Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

«Затверджую»
в.о. завідуючого кафедри
комп'ютерних систем та робототехніки
_____к. ф.-м. н., доцент Максим ХРУСЛОВ
«___» червня 2025 р.

Пояснювальна записка

до кваліфікаційної роботи
бакалавра

на тему: «МОДЕЛЬ ПРОГНОЗУВАННЯ СОЦІАЛЬНО-ЕКОНОМІЧНИХ ПОКАЗНИКІВ РОЗВИТКУ ДЕРЖАВИ»

Спеціальність 151 – Автоматизація, комп'ютерно-інтегровані технології
Галузь знань 15 – Автоматизація та приладобудування
Освітня програма «Автоматизація та комп'ютерно-інтегровані технології»

Захищено на засіданні Екзаменаційної комісії № 46
протокол № __від __.06.2025 р.
Оцінка ____/ _____
Голова Екзаменаційної комісії
_____**ЧУГАЙ А. М.**

Виконав:
Студент групи КУ– 41
ПОНОМАРЕНКО Віталій Володимирович _____

Керівник:
к.т.н., доцент кафедри комп'ютерних систем та робототехніки
СТРІЛЕЦЬ Вікторія Євгенівна _____

Рецензент: к.т.н., доцент, доцент кафедри математичного моделювання та аналізу даних
КОРОБЧИНСЬКИЙ Кирил Петрович _____

Харків – 2025

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи – 58 сторінки, з яких 50 сторінок становить основна частина, ілюстрована 18 рисунками та 4 таблицями. Список використаних джерел містить 33 найменувань.

Мета роботи – підвищити точність короткострокового прогнозування макроекономічних показників розвитку держав – ВВП, безробіття, інфляції та чисельності населення, через використання підходів машинного навчання.

Об'єкт дослідження – процес прогнозування макроекономічних показників розвитку держав.

Предмет дослідження – моделі і методи машинного навчання для прогнозування часових рядів, зокрема градієнтний бустинг на деревах XGBoost і рекурентна нейронна мережа Bidirectional LSTM, а також алгоритми їх гіперпараметричної оптимізації.

Актуальність теми полягає в необхідності оперативних і достовірних економічних прогнозів, які є підґрунтям для прийняття рішень у соціальній політиці. Недостатня точність статистичних методів при високій волатильності показників обумовлює потребу у використанні сучасних алгоритмів штучного інтелекту.

У роботі узагальнено класичні економетричні та ML-підходи. Побудовано конвеєр підготовки WDI-даних (очищення, лаги, нормування). Реалізовано навчання XGBoost (GridSearchCV + TimeSeriesSplit) і Bidirectional LSTM (EarlyStopping) в PyCharm. Оцінено моделі на єдиній вибірці за RMSE та MAE. Показано, що XGBoost знижує похибку у 1,5-8 разів.

Ключові слова: ПРОГНОЗУВАННЯ, СОЦІАЛЬНО-ЕКОНОМІЧНІ ПОКАЗНИКИ, ВВП, БЕЗРОБІТТЯ, ІНФЛЯЦІЯ, НАСЕЛЕННЯ, XGBOOST, LSTM, МАШИННЕ НАВЧАННЯ, ЧАСОВІ РЯДИ, RMSE, MAE.

ABSTRACT

The explanatory note to the bachelor's thesis comprises an introduction, three chapters, conclusions, a list of references, and appendices. The total length is 58 pages, of which 50 pages form the main body, illustrated by 18 figures and 4 tables. The reference list includes 33 entries.

Objective: To improve the accuracy of short-term forecasting of macroeconomic development indicators—GDP, unemployment, inflation, and population—through the use of machine-learning approaches.

Research object: The process of forecasting macroeconomic development indicators of states.

Research subject: Machine-learning models and methods for time-series forecasting, in particular the XGBoost gradient-boosted tree algorithm and a Bidirectional LSTM recurrent neural network, as well as their hyperparameter optimization algorithms.

Relevance: There is a critical need for timely and reliable economic forecasts to inform decision-making in social policy. The insufficient accuracy of traditional statistical methods under high indicator volatility necessitates the adoption of modern artificial-intelligence algorithms.

In this work, classical econometric and ML approaches are reviewed. A data-preprocessing pipeline for the World Development Indicators dataset (cleaning, lag generation, normalization) is constructed. Training of XGBoost (with GridSearchCV and TimeSeriesSplit) and Bidirectional LSTM (with EarlyStopping) is implemented in PyCharm. Both models are evaluated on a single dataset using RMSE and MAE. It is shown that XGBoost reduces forecasting error by a factor of 1.58.

***Keywords*:** FORECASTING, SOCIO-ECONOMIC INDICATORS, GDP, UNEMPLOYMENT, INFLATION, POPULATION, XGBOOST, LSTM, MACHINE LEARNING, TIME SERIES, RMSE, MAE.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

WDI	–	*World Development Indicators* – глобальна статистична база даних Світового банку
GDP	–	*Gross Domestic Product* – валовий внутрішній продукт
CPI	–	*Consumer Price Index* – індекс споживчих цін
RMSE	–	*Root Mean Square Error* – корінь середньоквадратичної помилки
MAE	–	*Mean Absolute Error* – середня абсолютна помилка
MSE	–	*Mean Squared Error* – середньоквадратична помилка
MAPE	–	*Mean Absolute Percentage Error* – середня абсолютна відносна помилка
R^2	–	*Coefficient of Determination* – коефіцієнт детермінації
ARIMA	–	*AutoRegressive Integrated Moving Average* – авторегресивна інтегрована рухома середня
VAR	–	*Vector AutoRegression* – векторна авторегресійна модель
RNN	–	*Recurrent Neural Network* – рекурентна нейронна мережа
LSTM	—	*Long Short-Term Memory* – нейронна мережа довготривалої короткочасної пам’яті
XGBoost	–	*Extreme Gradient Boosting* – алгоритм градієнтного бустингу дерев рішень

ВСТУП

Актуальність дослідження. Сучасна економічна політика ґрунтується на швидких і точних прогнозах макроекономічних показників – таких, як валовий внутрішній продукт (ВВП), рівень безробіття, інфляція та чисельність населення. Помилки у передбаченні цих величин призводять до неефективного бюджетного планування, хибних рішень щодо грошово-кредитної політики та соціальних програм. Класичні статистичні моделі (ARIMA, VAR тощо), хоч і залишаються корисними, часто не забезпечують достатньої точності за умов високої волатильності та нелінійності економічних процесів. Поява потужних методів машинного навчання відкриває нові можливості для підвищення якості прогнозів завдяки здатності алгоритмів ураховувати складні закономірності та великі обсяги даних.

Об’єкт дослідження – процес прогнозування макроекономічних показників розвитку держав.

Предмет дослідження – моделі і методи машинного навчання для прогнозування часових рядів, зокрема градієнтний бустинг на деревах XGBoost і рекурентна нейронна мережа Bidirectional LSTM, а також алгоритми їх гіперпараметричної оптимізації.

Мета роботи – підвищити точність короткострокового прогнозування макроекономічних показників розвитку держав – ВВП, безробіття, інфляції та чисельності населення, через використання підходів машинного навчання.

Завдання дослідження

1. Проаналізувати теоретичні положення класичних економетричних і сучасних ML-підходів до прогнозування часових рядів.

2. Сформувати набір даних для аналізу, а саме часові ряди макроекономічних показників, на основі даних, опублікованих у базі *World Development Indicators* Світового банку.

3. Сформувати конвеєр обробки WDI-даних – очищення, побудову лагових ознак, нормування.

4. Розробити та реалізувати в середовищі PyCharm програмні модулі навчання XGBoost (із GridSearchCV + TimeSeriesSplit) та Bidirectional LSTM (із EarlyStopping).

5. Провести експериментальне тестування моделей, обчислити метрики ефективності **RMSE** та **MAE** й побудувати порівняльні графіки «факт / прогноз».

6. Оцінити переваги та обмеження кожної моделі; сформулювати рекомендації щодо подальшого вдосконалення системи прогнозування.

Методи дослідження: аналіз наукових джерел; програмна обробка даних *WDI* мовою Python; побудова лагових ознак; машинне навчання (GridSearchCV, EarlyStopping); статистичний аналіз похибок (RMSE, MAE); візуалізація результатів.

Практичне значення роботи

Запропоновані моделі можна інтегрувати в аналітичні дашборди центральних і регіональних органів влади для оперативного сценарного аналізу. Використання XGBoost дає змогу знизити середню квадратичну похибку прогнозу макропоказників у 1,5 – 8 разів порівняно з LSTM, що підвищує надійність бюджетних та монетарних рішень. Розроблений програмний код є reproducible-прототипом, придатним для розширення на багатовимірні або більш тривалі горизонти прогнозу.

РОЗДІЛ 1
АНАЛІЗ МОДЕЛЕЙ І МЕТОДІВ ПРОГНОЗУВАННЯ СОЦІАЛЬНО-ЕКОНОМІЧНИХ ПОКАЗНИКІВ

1.1 Соціально-економічні показники та їх роль у розвитку держав

Соціально-економічні показники – це кількісні індикатори, що характеризують стан економіки та добробут суспільства. До ключових показників належать валовий внутрішній продукт (ВВП), рівень інфляції, рівень безробіття та чисельність населення. Ці величини широко використовуються в державному управлінні для аналізу економічного розвитку, порівняння між країнами та ухвалення політичних рішень. Вони відображають різні аспекти макроекономічного стану держави і слугують базою для формування економічної політики [24, 4].

Валовий внутрішній продукт (ВВП) – це ринкова вартість усіх кінцевих товарів і послуг, вироблених у межах країни за певний період (як правило, за рік). ВВП є основним узагальнюючим показником економічної активності: він відображає обсяг національного виробництва та темпи економічного зростання. У державному управлінні динаміка реального ВВП (скоригованого на інфляцію) використовується для оцінки економічного розвитку, планування бюджету та визначення ефективності економічної політики [4, 13]. Крім того, ВВП на душу населення слугує індикатором середнього рівня добробуту громадян і часто використовується для міжнародних порівнянь рівня розвитку держав. Зростання ВВП зазвичай асоціюється з розширенням економічних можливостей, збільшенням доходів і покращенням соціальних стандартів життя [24].

Інфляція – це стійке загальне зростання рівня цін на товари та послуги в економіці протягом часу, що веде до знецінення грошової одиниці. Іншими словами, інфляція відображає, наскільки в середньому зросли ціни за рік. Помірна інфляція є нормальним явищем для зростаючої економіки, але високі темпи інфляції небезпечні, адже знижують реальні доходи населення та породжують економічну невизначеність [13, 25]. Відтак рівень інфляції є

одним з ключових орієнтирів для центробанків і урядів: надмірна інфляція загрожує економічній стабільності (відомий факт – у 1970-х роках інфляцію оголошували «ворогом №1» економіки США). Тому центральні банки (як-от НБУ) встановлюють цільові показники інфляції і проводять монетарну політику, спрямовану на підтримання стабільності цін [25]. Контрольована низька інфляція сприяє довгостроковому економічному зростанню, тоді як гіперінфляція або дефляція (зниження загального рівня цін) можуть призводити до кризових явищ.

Безробіття характеризує частку економічно активного населення, яке не має роботи, але активно її шукає та готове приступити до роботи. Рівень безробіття вимірюється як відсоток безробітних від загальної робочої сили. Цей показник відображає стан ринку праці: високий рівень безробіття сигналізує про економічний спад або структурні проблеми, тоді як низький – про наближення економіки до повної зайнятості. Для уряду важливо контролювати безробіття, адже воно напряму впливає на добробут населення, обсяги виробництва (через недовикористання трудових ресурсів) та бюджетні витрати (на програми допомоги по безробіттю тощо).

Макроекономічна теорія пов’язує коливання безробіття з економічним циклом і інфляцією: наприклад, крива Філіпса відображає обернену залежність між безробіттям та інфляцією у короткостроковому періоді [14]. Тому державна політика прагне знизити безробіття до так званого природного рівня – коли є лише фрикційне та структурне безробіття, але відсутнє циклічне (спадове) безробіття. Згідно з методологією Міжнародної організації праці, до безробітних відносять людей працездатного віку, які не мають роботи, активно шукають роботу та готові приступити до неї найближчим часом [19]. Відповідно, рівень безробіття є важливим індикатором соціально-економічного благополуччя і використовується для оцінки ефективності економічної політики (наприклад, через закон Оукена, який пов’язує зростання ВВП із зниженням безробіття) [5].

Чисельність населення та демографічна структура – фундаментальні соціально-економічні показники, що визначають масштаби ринку праці, споживчий попит та виробничий потенціал країни. Відоме висловлювання «демографія – це доля» підкреслює, що розмір, приріст і вікова структура населення багато в чому визначають довгостроковий соціально-економічний розвиток держави. Зростання чисельності населення, за інших рівних умов, розширює економіку (збільшується трудовий ресурс і кількість споживачів). Молоде зростаюче населення може стати «демографічним дивідендом» – потужним рушієм економічного зростання за умови забезпечення його роботою та освітою [3]. Натомість скорочення або старіння населення створює виклики: підвищується навантаження на працююче населення щодо утримання пенсіонерів, може бракувати робочої сили в перспективних галузях.

Державне планування обов'язково враховує демографічні показники – розробляються стратегії розвитку людського капіталу, системи охорони здоров'я, освіти та пенсійного забезпечення з огляду на прогнози змін чисельності та структури населення. Наприклад, для оцінки рівня економічного розвитку країни ВВП завжди співвідносять із чисельністю населення (обчислюючи ВВП на душу населення) [24], а при аналізі продуктивності праці беруть до уваги частку працездатного населення. Таким чином, населення є базисним показником, що впливає на більшість інших соціально-економічних індикаторів.

Отже, ВВП, інфляція, безробіття та населення виступають ключовими соціально-економічними показниками. *Їхня роль як показників розвитку держав* полягає в тому, що вони відображають результативність економіки та якість життя, а отже, служать орієнтирами для формування державної політики. Уряди країн, аналізуючи ці індикатори, визначають пріоритети макроекономічного регулювання: стимулювання економічного зростання, підтримання цінової стабільності, забезпечення зайнятості та розвиток людського потенціалу. Наприклад, за прискорення інфляції центральний банк

підвищує облікову ставку, при зростанні безробіття уряд впроваджує програми створення робочих місць, а при низьких темпах зростання ВВП розглядаються стимули для інвестицій та споживання. Таким чином, система соціально-економічних показників є своєрідним «приладовим щитком» економіки держави, за яким стежать політики й економісти, щоб своєчасно реагувати на виклики та спрямовувати розвиток у бажане русло.

1.2 Огляд методів прогнозування часових рядів

Прогнозування соціально-економічних показників за допомогою часових рядів є важливим завданням економічного аналізу. Часовий ряд – це послідовність значень певного показника, впорядкованих у хронологічному порядку (наприклад, квартальні значення ВВП або місячні значення інфляції) [5]. Моделі прогнозування часових рядів намагаються виявити патерни в минулих даних (тренди, сезонність, циклічність, автокореляції) і екстраполювати їх у майбутнє. Існує широкий спектр методів – від класичних статистичних моделей до сучасних методів машинного навчання та глибоких нейронних мереж. Кожен з підходів має свої переваги та обмеження, які доцільно розглянути для обґрунтування вибору моделі прогнозування [28, 32].

На рис. 1.1 зображено приклади типових форм часових рядів: горизонтальний (сталий) ряд без вираженого тренду, ряд з тенденцією зростання (трендом), а також ряд з трендом і сезонними коливаннями. Ці компоненти – рівень, тренд і сезонність – часто враховуються при моделюванні, оскільки багато економічних показників мають довгострокові тренди (наприклад, зростання ВВП) та сезонні коливання (наприклад, збільшення споживання енергії взимку). У реальних економічних даних також присутня випадкова компонента (шум), що ускладнює точний прогноз. Завдання моделей прогнозування – відфільтрувати шум і виділити структурні закономірності, щоб передбачити майбутні значення показників з мінімальною похибкою.

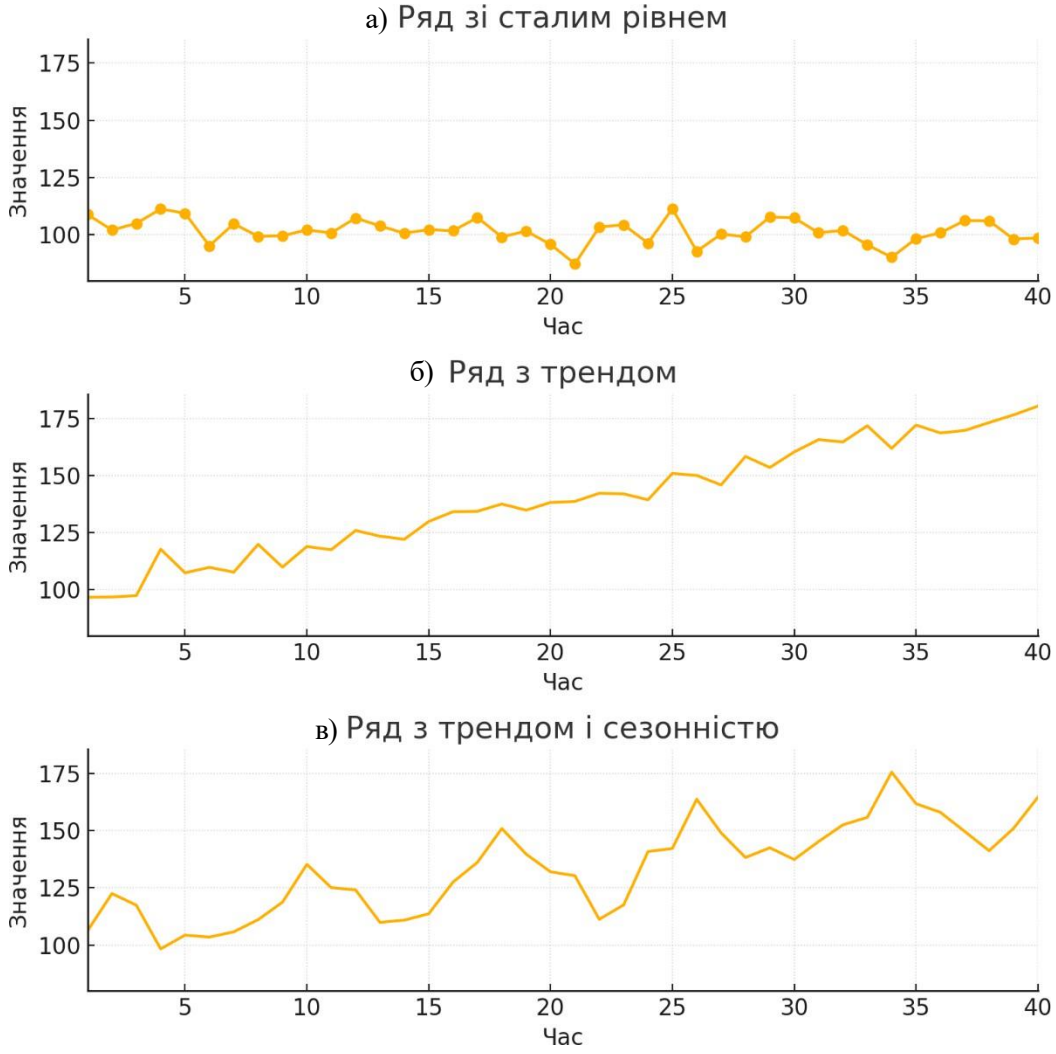

Рисунок 1.1 – Приклади часових рядів: а) ряд зі сталим рівнем (без тренду), б) ряд з висхідним трендом, в) ряд з трендом і сезонністю

Класичні статистичні моделі прогнозування. Традиційним підходом до аналізу часових рядів є використання лінійних моделей, заснованих на автокореляції. Найбільш розповсюдженою є модель **ARIMA** (Autoregressive Integrated Moving Average) – авторегресійна інтегрована середня ковзна модель(Box–Jenkins методологія) [2,11]. ARIMA поєднує в собі: авторегресію (AR) – врахування залежності поточного значення ряду від попередніх значень, інтегрованість (I) – різницювання ряду для досягнення стаціонарност, та середнє ковзне (MA) – врахування впливу попередніх випадкових шумів (помилок). Модель ARIMA позначається як ARIMA (p, d, q), де p – порядок

авторегресії, d – порядок різницювання, q – порядок середнього ковзного. Формально ARIMA (p, d, q) задається рівнянням виду:

$$x_t = \alpha + \boldsymbol{\phi}_1 x_{t-1} + \cdots + \boldsymbol{\phi}_p x_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t, \qquad (1.1)$$

де x_t – значення ряду в момент t, ε_t – випадкова помилка (шум) в момент t, $\boldsymbol{\phi}_i$ – коефіцієнти авторегресії, θ_j – коефіцієнти середнього ковзного, а α – константа. Оператор інтегрування порядку d означає, що модель будується не для самого ряду, а для його d-ї різниці $\nabla^d x_t$ (тобто застосовано різницювання для усунення тренду або сезонності, щоб ряд став стаціонарним). Модель ARIMA підбирається шляхом аналізу автокореляційної функції (ACF) та часткової автокореляції, з подальшим оцінюванням параметрів методом найменших квадратів або максимального правдоподібності.

Перевагами ARIMA є її простота та інтерпретованість: коефіцієнти $\boldsymbol{\phi}_i$ показують вплив попередніх значень, а θ_j – вплив попередніх шоків на поточне значення. ARIMA добре захоплює лінійні тренди та сезонні компоненти (через різницювання і розширення до SARIMA для сезонних ефектів) і за достатньо коротких прогнозних горизонтів часто дає точні результати. Вона ефективна на відносно невеликих вибірках даних і не потребує великої кількості параметрів для оцінки (при невеликих p та q).

Втім, обмеження ARIMA полягають у тому, що вона є суто лінійною моделлю і не враховує можливих нелінійностей у динаміці показників. ARIMA вимагає стаціонарності ряду – якщо процес має зміни дисперсії або структурні зрушення, модель може давати хибні результати [12]. Крім того, ARIMA є одновимірною (уніваріантною) моделлю, тобто враховує лише власні попередні значення ряду і не дозволяє прямо включити до моделі інші пов’язані показники [22].

Для багатовимірного прогнозування застосовується розширення – **VAR** (Vector Autoregression, векторна авторегресія). Модель VAR – це система рівнянь, де кожен з декількох часових рядів моделюється як авторегресія на лагові значення *всіх* включених перемінних. Наприклад, для одночасного

прогнозування ВВП, інфляції та безробіття будуть побудовані три рівняння, в яких значення ВВП залежить від попередніх значень і ВВП, і інфляції, і безробіття; аналогічно для інших двох показників. Таким чином, VAR-моделі здатні врахувати взаємозв'язки між економічними показниками (наприклад, вплив інфляції на ВВП або безробіття на інфляцію) без потреби задавати наперед причинно-наслідкову структуру – всі змінні трактуються як ендогенні. Це робить VAR гнучким інструментом для аналізу складних економічних систем. Зокрема, на основі VAR можна будувати імпульсні відгуки та оцінювати, як шок в одній змінній вплине на інші з часом.

Переваги VAR: відносна простота оцінювання (можна оцінювати покомпонентно методами регресії), можливість охопити багатомірні залежності та хороші прогнозні властивості при наявності довгої історії спостережень. Однак суттєвим недоліком VAR є велика кількість параметрів, які потрібно оцінити. Якщо модель включає K змінних і p лагів, то загальна кількість параметрів порядку $(K + pK^2)$, що швидко зростає зі збільшенням числа змінних. Для коректної оцінки такої моделі потрібна дуже довга вибірка даних; інакше є ризик перенавчання (overfitting) та нестабільності оцінок. Крім того, VAR – суто статистична модель (її інколи називають «a-theoretical»), тобто вона не ґрунтується на економічній теорії причинно-наслідкових зв'язків. Це ускладнює інтерпретацію результатів: модель показує кореляційні взаємозв'язки, але не пояснює, *чому* одна змінна впливає на іншу. Тим не менш, VAR широко використовується економістами, зокрема в центральних банках, для короткострокового прогнозування макропоказників і аналізу шоків, коли є достатньо довгі часові ряди даних.

Методи машинного навчання. З 2000-х років в економічне прогнозування активно впроваджуються методи машинного навчання (Machine Learning, ML)[20], які добре себе зарекомендували в задачах класифікації та регресії на табличних даних. До таких методів належать, зокрема, алгоритми на основі дерев рішень – Random Forest та XGBoost. Їх застосування до часових рядів зазвичай передбачає перетворення вихідного

ряду на набір лагових ознак (факторів). Тобто, замість прямого авторегресійного рівняння моделюється залежність y_{t+h} (значення ряду у майбутньому через h періодів) від значень y_t, y_{t-1} … , y_{t-p}, та, за потреби, інших пояснювальних змінних. Таким чином, задачу прогнозування часових рядів зводять до завдання машинного навчання: на вхід моделі подаються ознаки (лаги ряду, можливо інші змінні), а на виході – прогнозоване значення. Алгоритми Random Forest та XGBoost можуть виявляти складні нелінійні залежності між ознаками і цільовою змінною, що дає їм потенційну перевагу над лінійними моделями на кшталт ARIMA в разі, коли взаємозв'язки в даних нелінійні або коли до прогнозу залучено багато додаткових факторів.

Random Forest (випадковий ліс) – це ансамблевий метод на основі дерева рішень. Алгоритм будує велику кількість дерев рішень на різних випадкових підвибірках даних і з різним випадковим піднабором ознак, а потім усереднює результати цих дерев (шляхом голосування для класифікації або усереднення для регресії). Така ансамблева схема дозволяє знизити варіативність прогнозу і уникнути перенавчання, яке притаманне окремим глибоким деревам рішень. Кожне дерево у випадковому лісі навчиться певним аспектам залежності, а усереднення згладить шум і дасть більш стійкий прогноз. У випадку прогнозування часового ряду, Random Forest фактично будує різні рішення, як пов'язані минулі значення перетворюються на майбутні, і комбінує ці рішення [6].

Перевагою Random Forest є його *гнучкість і стійкість до шуму*: він може врахувати нелінійні ефекти, взаємодії між лагами та навіть аномальні ситуації, оскільки окремі дерева здатні фокусуватися на різних частинах простору ознак. На практиці Random Forest показав високу точність у багатьох завданнях і часто перевершує традиційні моделі, коли в розпорядженні достатньо великий масив даних. Наприклад, в одному з досліджень було продемонстровано, що модель Random Forest забезпечила кращу прогностичну точність, ніж ARIMA, при передбаченні спалахів захворювання (грипу) на основі часових рядів інцидентності. У цьому випадку випадковий

ліс зміг врахувати нелінійні патерни в епідеміологічних даних і дав більш точний прогноз, ніж класична статистична модель.

Обмеження Random Forest – *низька інтерпретованість* моделі та потреба у достатньому обсязі даних. На відміну від ARIMA, де можна явно виписати вплив попередніх спостережень через коефіцієнти, у випадковому лісі прогноз формується колективним рішенням сотень дерев, що ускладнює пояснення, чому модель зробила той чи інший прогноз для конкретного моменту. Хоча існують методи оцінки важливості ознак у Random Forest, вони дають лише загальне уявлення, які лаги чи фактори найбільш впливові, але не надають простої аналітичної формули зв'язку. Крім того, для навчання надійного випадкового лісу бажано мати великий набір тренувальних спостережень, оскільки модель містить багато параметрів (структури дерев) і може переобладнатися на малих вибірках. У випадку коротких часових рядів класичні моделі можуть виявитися точнішими саме через більш стриману параметризацію. В цілому, Random Forest є ефективним інструментом, особливо коли економічний процес залежить від багатьох факторів і спостерігається довга історія даних.

XGBoost (Extreme Gradient Boosting) – ще один популярний алгоритм машинного навчання на основі дерев рішень, який реалізує метод градієнтного бустингу. На відміну від випадкового лісу, де дерева будуються паралельно і незалежно, у бустингу дерева будуються послідовно, кожне наступне дерево намагається виправити помилки попередніх. Алгоритм XGBoost починає з простого дерева і поступово додає нові дерева, які моделюють залишкові помилки (градієнти) попереднього ансамблю. Завдяки цьому модель поступово покращує точність, концентруючись на складних для прогнозу випадках. XGBoost відзначається високою ефективністю реалізації: алгоритм оптимізований для обчислювальної швидкості та має вбудовані засоби регуляризації для уникнення перенавчання. Зокрема, цільова функція XGBoost містить штраф за складність моделі (кількість листів дерев, їх глибину тощо), що допомагає тримати модель «в узді» навіть за великої кількості ітерацій

бустингу. Також XGBoost використовує другу похідну (кривизну) помилки при побудові дерев, що забезпечує більш точний рух у бік градієнту і швидшу збіжність алгоритму. На практиці XGBoost неодноразово демонстрував найвищу точність прогнозів на різних змаганнях з аналізу даних, зокрема і для часових рядів, коли дані належним чином підготовлені.

При застосуванні до часових рядів XGBoost так само потребує вибору ознак (лагів, можливих зовнішніх регресорів) і не має явного механізму врахування порядку часу, окрім як через ці ознаки. Але завдяки своїй потужності XGBoost може виявляти і складні нелінійні комбінації лагових значень. Дослідження показують, що XGBoost здатен випереджати навіть нейронні мережі в деяких задачах прогнозування, особливо коли часовий ряд піддається ефективному попередньому опрацюванню. Переваги XGBoost – це *висока точність і продуктивність*: алгоритм використовує ресурси апаратного забезпечення максимально ефективно (паралелізація, обтинання дерева за принципом жадібного пошуку) і може працювати з великими масивами даних. Він також більш контрольований: параметри регуляризації дозволяють запобігти перенавчанню навіть за великої глибини дерев.

Недоліки XGBoost подібні до Random Forest: модель бустингу на деревах є *складною для інтерпретації* (ще менш прозорою, ніж випадковий ліс, через послідовну взаємодію дерев). Аналітику важко витлумачити, як окремі лаги впливають на прогноз, оскільки ефект розподілений через багаторівневу композицію багатьох дрібних дерев. Окрім того, бустинг вимагає уважного налаштування гіперпараметрів (кількість дерев, швидкість навчання, глибина дерев тощо) і достатнього обсягу даних для їх надійного визначення. При малих вибірках XGBoost може перенавчитись, якщо не застосувати адекватну регуляризацію. Але загалом XGBoost є одним з найпотужніших алгоритмів прогнозування і часто використовується як еталон для порівняння нових методів.

Нейронні мережі. Окрему нішу в прогнозуванні часових рядів займають моделі глибокого навчання, зокрема **рекурентні нейронні мережі (RNN)** та їх

підвид – **довгокороткострокова пам'ять (LSTM)**. На відміну від статистичних моделей і традиційних ML-алгоритмів, розглянутих вище, нейронні мережі здатні автоматично виокремлювати складні шаблони з даних і набагато краще пристосовані до моделювання нелінійної динаміки.

Рекурентна нейронна мережа (RNN) – це архітектура штучної нейронної мережі, спеціально призначена для послідовних даних. Ключова ідея RNN – *наявність зворотних зв'язків (рекурентності)*, які дозволяють мережі запам'ятовувати інформацію про попередні елементи послідовності при обробці наступних. Структурно RNN складається з шарів нейронів, кожен з яких на кожному кроці часу приймає на вхід не тільки поточний сигнал x_t, але й свій власний стан з попереднього кроку (прихований стан h_{t-1}). Таким чином, вихід мережі y_t на кроці t залежить як від x_t, так і від історії x_{t-1}, x_{t-2}, ... закодованої у прихованому стані (рис. 1.2).

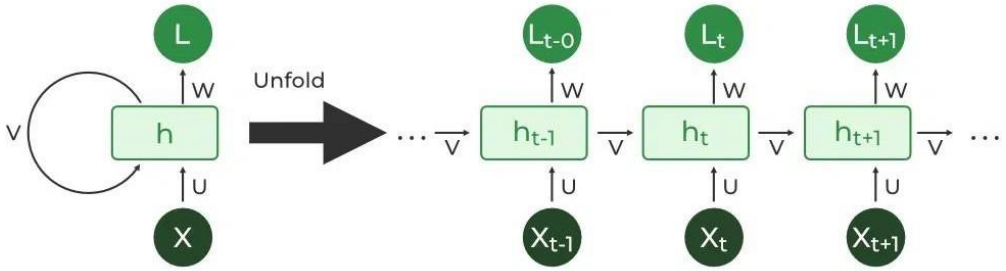

Рисунок 1.2 – Схема рекурентної нейронної мережі: зворотній зв'язок прихованого стану h дозволяє "згортати" послідовність, запам'ятовуючи попередню інформацію при обробці нових даних

На практиці це реалізується як "розгортання" мережі в часі: RNN можна уявити як багато копій звичайної нейронної мережі, зв'язаних послідовно, де кожна передає наступній своє приховане представлення (стан). Така рекурентна архітектура дозволяє мережі накопичувати знання про попередні значення ряду і використовувати контекст для прогнозування майбутнього.

RNN-моделі в теорії можуть враховувати довільно довгу історію послідовності. Однак на практиці базові RNN стикаються з проблемою *згасаючого градієнта* при навчанні: під час алгоритму зворотного поширення помилки через час (Backpropagation Through Time) вплив далеких кроків минулого експоненціально зменшується, і мережа «забуває» далеку історію, фокусуючись лише на недавніх значеннях. Це виявилося суттєвою перешкодою для застосування RNN в економіці, де важливі довгострокові залежності (напр. ефект демографічних змін на економіку розгортається десятиліттями). Рішенням стала модифікована архітектура **LSTM (Long Short-Term Memory)**, запропонована Шмідхубером і Гохрайтером у 1997 році. Мережа LSTM вводить спеціальні механізми – *«гейти» (ворота)*, що керують потоком інформації: вхідний гейт вирішує, яку нову інформацію додати до стану, гейт забування – яку інформацію стерти, а вихідний гейт – яку інформацію використати для розрахунку виходу. Завдяки цьому LSTM може утримувати значущі сигнали у своєму внутрішньому стані протягом дуже довгого часу, фактично долаючи проблему згасаючого градієнта. LSTM-мережі *здатні вчитися довгострокових залежностей*: наприклад, розпізнавати, що певні економічні цикли повторюються кожні N років, або що наслідки фінансової кризи впливатимуть на економіку протягом кількох наступних періодів. Саме ця здатність робить LSTM надзвичайно привабливою для задач економічного прогнозування з довгою пам’яттю. За останні роки LSTM набули популярності в задачах прогнозування фінансових та економічних часових рядів, зокрема для прогнозування цін на криптовалюту, біржових індексів, попиту на електроенергію тощо [17].

Переваги нейронних мереж (RNN/LSTM) у прогнозуванні часових рядів: по-перше, *гнучкість та нелінійність* – вони можуть апроксимувати надскладні функції залежності і вловлювати патерни, недоступні лінійним моделям. По-друге, *автоматичне виявлення ознак* – на відміну від моделей на основі дерев, яким потрібні явні лагові ознаки, RNN самі «витягують» необхідну інформацію з послідовності через навчання своїх ваг. По-третє,

врахування довгих лагів – LSTM спеціально сконструйовані для утримання інформації про далеке минуле, тому можуть моделювати процеси з довгою пам'яттю. У порівнянні з ARIMA, яка враховує обмежене число лагів p, нейронна мережа теоретично може задіяти набагато більший горизонт історії (лише б дані це підтримували) [8,16].

Втім, у нейромереж теж є суттєві *недоліки*. По-перше, це *вимоги до даних*: для навчання глибокої мережі потрібна велика кількість історичних спостережень, інакше модель може переналаштуватись під шум. Часові ряди макроекономічних показників часто є відносно короткими (кілька десятків років поквартально – це лише близько 200 точок), що ускладнює ефективне навчання нейронної мережі без спеціальних прийомів (доприклад, передтренування або використання даних з інших країн). По-друге, нейронні моделі потребують значних обчислювальних ресурсів (час навчання, підбір гіперпараметрів). На відміну від ARIMA чи навіть Random Forest, які можна швидко оцінити на звичайному комп'ютері, навчання LSTM вимагає більше часу і часто прискорюється на графічних процесорах (GPU). По-третє, і це критично для застосування в державному управлінні, – *нейронні мережі малопридатні до інтерпретації*. Вони діють як «чорний ящик»: важко пояснити, як саме комбінація ваг у багатошаровій нейронній мережі генерує те чи інше прогнозне значення. Для економістів і посадовців, які звикли до прозорих моделей (наприклад, регресій), це є суттєвим бар'єром у довірі до прогнозу. В літературі наголошується, що коли головною вимогою є інтерпретованість і прозорість, традиційні методи часових рядів залишаються кращими саме через зрозумілість їх структури. З іншого боку, якщо стоїть пріоритет максимальної точності і є достатньо даних та обчислювальних потужностей, нейронні мережі можуть дати виграш у якості прогнозу.

Підсумовуючи, розглянемо **порівняння різних моделей** за кількома важливими критеріями (табл. 1). Критерії включають інтерпретованість моделі, типово досяжну точність прогнозу, потребу в обсязі даних для

навчання, здатність моделі враховувати великі лаги (довгострокові залежності) та обчислювальну складність.

Таблиця 1.1

Порівняння моделей прогнозування часових рядів за основними характеристиками

Модель	Інтерпрето-ваність	Точність	Потреба в даних	Облік довгих лагів	Обчислю-вальна складність
ARIMA	Висока	Помірна для лінійних трендів	Низька (ефективна на малих вибірках)	Обмежений лаг p	Низька (швидка оцінка)
VAR	Середня	Висока (за умови достатніх даних)	Дуже висока (довгі ряди)	Обмежений лаг p (багатовимірний)	Вища (багато параметрів)
Random Forest	Низька	Висока (на нелінійних даних)	Середня	Залежить від максимально введеного лагу	Середня (паралельне навчання)
XGBoost	Низька	Дуже висока	Висока	Залежить від введених лагів	Висока (оптимізо-вана)
RNN (LSTM)	Низька	Дуже висока (для складних залежностей)	Дуже висока	Відмінна (вбудована довгострокова пам’ять)	Висока (тривале навчання)

- **ARIMA**: дуже висока інтерпретованість (лінійні параметри легко пояснити), помірна точність для лінійних та стаціонарних процесів, низькі вимоги до даних (ефективна навіть на десятках спостережень), обмежена довжина пам’яті (лише p останніх лагів напряму враховуються), дуже швидке оцінювання і прогнозування (низька обчислювальна вартість).
- **VAR**: середня інтерпретованість (можна аналізувати взаємодії змінних через коефіцієнти й імпульсні відгуки, але модель громіздка), хороша точність при наявності багатьох даних і взаємопов’язаних показників, дуже високі вимоги до даних (потрібні довгі часові ряди для надійної оцінки параметрів), модель враховує взаємозв’язки і лаги по всіх змінних (довша

пам’ять, ніж у ARIMA, але теж обмежена лагом p), обчислювально більш витратна (оцінювання великої кількості параметрів, особливо при Bayesian VAR або TVP-VAR, потребує значних ресурсів).

- **Random Forest**: низька інтерпретованість (результат агрегування багатьох дерев складно пояснити у термінах коефіцієнтів), висока точність на нелінійних даних або за наявності багатьох ознак, середні вимоги до даних (потрібно більше, ніж для ARIMA, але за рахунок усереднення може працювати і на помірних вибірках), неявно може враховувати досить довгі лаги, якщо їх подати як ознаки (пам’ять визначається вибором максимального лагу при підготовці даних), обчислювальна вартість – середня (можна навчати паралельно багато дерев, масштабування на багато ядер добре вирішує проблему).
- **XGBoost**: низька інтерпретованість (як і RF, бустинг з дерев мало прозорий для аналізу), дуже висока точність (часто одна з найкращих на змагальних задачах за рахунок поєднання багатьох нелінійних залежностей), високі вимоги до даних (для уникнення перенавчання і для навчання багатьох параметрів потрібна значна вибірка), врахування довгих лагів також через подання відповідних ознак (немає внутрішньої пам’яті, але можна додати скільки завгодно лагів як вхід), обчислювально інтенсивний (високе навантаження, але алгоритм оптимізований і використовує апаратне прискорення, тому навчання досить швидке навіть на великому обсязі даних, проте послідовний характер бустингу не так легко паралелізувати, як Random Forest) [15].
- **RNN/LSTM**: низька інтерпретованість (нейронна мережа – «чорна скринька», важко отримати просте пояснення прогнозу), потенційно найвища точність при складних патернах (особливо LSTM при наявності нелінійностей і довгих залежностей може суттєво перевершувати інші моделі за якістю прогнозу), дуже високі вимоги до даних (для навчання нейронної мережі потрібно на порядок більше спостережень, ніж для ARIMA, бажано доповнювати іншими джерелами або генерувати синтетичні дані), здатність

враховувати довгі лаги – відмінна (LSTM спеціально сконструйована для довгострокової пам’яті і може зберігати інформацію про стан десятки і сотні кроків тому), обчислювальна складність – висока (тренування може тривати довго, вимагає GPU, підбір архітектури і гіперпараметрів також нетривіальний).

1.3 Вибір методів і моделей дослідження

На підставі аналізу методів прогнозування для дослідження обрано два підходи – градієнтний бустинг (XGBoost) та рекурентні нейронні мережі LSTM. Ці моделі представляють сучасні досягнення машинного та глибокого навчання, доповнюючи одне одного: XGBoost відзначається високою точністю у разі складних нелінійних залежностей і багатофакторності, а LSTM здатна враховувати довгострокову пам’ять часових рядів і виявляти приховані динамічні патерни. Обидві моделі вже застосовувалися в суміжних задачах і показали переваги над класичними методами, що дає підстави сподіватися на їх ефективність у прогнозуванні соціально-економічних показників [23].

Обґрунтування вибору XGBoost. XGBoost має здатність моделювати нелінійні взаємозв’язки серед численних вхідних факторів. Економічні показники (наприклад, ВВП чи інфляція) залежать від сукупності причин: інвестицій, споживання, зовнішніх шоків тощо. Деревоподібні моделі природно працюють із великою кількістю ознак, відбираючи найінформативніші розгалуження. Це важливо, коли потрібно врахувати максимум доступних даних (історичні ряди, супутні індикатори, ознаки криз чи війни). До того ж XGBoost стійкий до шуму й перенавчання завдяки вбудованій регуляризації, що є перевагою в умовах економічних коливань. Також він ефективний з обчислювальної точки зору: швидко навчається навіть на великих вибірках і дозволяє проводити багаторазову перехресну перевірку моделей. Отже, XGBoost обрано як потужний і репрезентативний метод машинного навчання для точного прогнозування.

Обґрунтування вибору LSTM. LSTM обрана через здатність зберігати інформацію про минулі стани впродовж тривалого часу та моделювати складні

часові залежності. В економічних задачах часто спостерігаються довгострокові тренди (наприклад поступове сповільнення зростання), затримані ефекти політичних рішень або циклічні коливання (бізнес-цикли 5–10 років). Традиційні моделі з обмеженою кількістю лагів можуть пропустити такі впливи, а механізм «воріт» у LSTM дозволяє накопичувати й поступово коригувати інформацію про подію протягом тривалого терміну. Таким чином, LSTM може вловити, наприклад, вплив демографічного вибуху через кілька років або затяжну рецесію на ВВП навіть після закінчення спаду. Крім того, нейромережа здатна моделювати високонелінійні комбінації циклів і трендів, що характерно для економічних процесів. Хоча LSTM вимагає ретельного налаштування й великих даних, передбачається використати всі доступні джерела та за потреби збільшити вибірку через агрегування різних частот або перенос знань із суміжних економік. Попередні дослідження підтверджують переваги LSTM над класичними моделями у низці економічних задач [33].

Висновки за розділом 1

Для розробки моделей прогнозування макроекономічних показників обрано дві моделі – XGBoost та LSTM – як такі, що відображають сучасні тенденції і доповнюють одна одну: XGBoost – більш інтерпретована для аналітика (через важливості ознак, хоча й не так прозора, як ARIMA) та швидка, LSTM – більш адаптивна до складної динаміки і довготривалих впливів. Комбінований підхід дозволить порівняти результати і зробити більш обґрунтовані висновки про придатність методів машинного навчання для прогнозування соціально-економічного розвитку.

Основне завдання дослідження – розробити та обґрунтувати модель прогнозування ключових соціально-економічних показників розвитку держав на основі методів машинного навчання (XGBoost) та глибокого навчання (LSTM), а також оцінити точність і практичну корисність цих підходів у порівнянні з класичними методами. Для досягнення поставленої мети необхідно вирішити такі *задачі*:

- проаналізувати роль основних соціально-економічних показників (ВВП, інфляція, безробіття, чисельність населення) у державному управлінні та обґрунтувати актуальність їх прогнозування;
- здійснити критичний огляд існуючих методів прогнозування часових рядів, висвітлити їх переваги та недоліки на основі наукових джерел;
- вибрати перспективні моделі для прогнозування (XGBoost, LSTM) та розробити методику їх застосування до економічних показників, включаючи підготовку даних та вибір гіперпараметрів;
- реалізувати обрані моделі на реальних даних соціально-економічних показників (для прикладу, макроекономічних індикаторів України чи інших держав) та провести експериментальне порівняння їх точності прогнозу з базовою статистичною моделлю (наприклад, ARIMA);
- проаналізувати результати експерименту, інтерпретувати поведінку моделей (в межах можливого, зокрема через оцінку важливостей ознак у XGBoost) та зробити висновки щодо доцільності використання кожного з підходів;
- сформулювати рекомендації щодо впровадження моделей прогнозування соціально-економічних показників у практику державного управління, враховуючи точність прогнозів та вимоги до інтерпретації.

Виконання зазначених завдань дозволить отримати цілісне уявлення про можливості сучасних моделей прогнозування у сфері макроекономіки. У результаті дослідження очікується розробка моделі або набору моделей, які забезпечують підвищення точності довгострокового прогнозу соціально-економічних показників розвитку держав порівняно з традиційними підходами, а також оцінка, наскільки такі моделі можуть бути інтегровані в процес ухвалення рішень органами влади. Зрештою, це сприятиме поліпшенню якості економічного планування та прогнозування на державному рівні, що особливо актуально в умовах невизначеності та швидких змін, притаманних сучасній світовій економіці.

РОЗДІЛ 2
МОДЕЛЬ ПРОГНОЗУВАННЯ СОЦІАЛЬНО-ЕКОНОМІЧНИХ ПОКАЗНИКІВ РОЗВИТКУ ДЕРЖАВ

2.1 Джерело та підготовка навчальних даних

Для побудови моделі було використано відкритий датасет *World Development Indicators (WDI)* [19,2`] від Світового банку – одну з найповніших баз даних соціально-економічних показників розвитку країн світу. Ця база містить близько 1600 показників для майже 220 країн та територій, причому ряд показників простежується протягом понад 50 років. У роботі з WDI було обрано чотири ключові індикатори, які відображають соціально-економічний розвиток держави:

- *валовий внутрішній продукт (ВВП)* – загальна вартість всіх товарів і послуг, вироблених у країні (у постійних цінах, для усунення впливу інфляції);
- *чисельність населення* – загальна кількість мешканців країни;
- *рівень безробіття* – частка економічно активного населення, що не має роботи (% робочої сили);
- *рівень інфляції* – річний темп зростання цін (%, наприклад індекс споживчих цін або дефлятор ВВП).

Вибір саме цих показників обумовлений тим, що вони комплексно характеризують економіку (ВВП), демографію (населення) та соціальну сферу (безробіття, інфляція) країни. Дані по кожному з обраних показників були завантажені з WDI у вигляді таблиці (CSV-файл WDI.csv), що містить часові ряди по роках для різних країн світу. Для дослідження було відібрано дані близько 180 країн за період 1990–2020 рр. – саме цей інтервал часу забезпечує наявність інформації по всіх чотирьох показниках для більшості держав (ранніші роки часто містять пропуски по безробіттю чи інфляції). Таким чином, фінальний набір даних являє собою панель: для кожної країни доступні значення ВВП, населення, безробіття та інфляції по роках в зазначеному діапазоні.

Перед використанням даних у моделях був проведений їх детальний аналіз і попередня обробка. Особливу увагу приділено пропущеним значенням, масштабуванню та формуванню вхідних послідовностей для моделей прогнозування.

Вибір цільових змінних. Цільовими змінними для прогнозування обрано одразу *кілька показників*: ВВП, чисельність населення, рівень безробіття та рівень інфляції. На відміну від задачі з однією вихідною змінною, тут розглядається *багатоваріантний прогноз* – модель одночасно передбачає майбутні значення всіх чотирьох індикаторів. Такий підхід дозволяє врахувати можливі взаємозв'язки між показниками (наприклад, вплив зростання населення на ВВП, вплив ВВП на безробіття тощо) при навчанні моделі. Вихідними (target) для моделі є значення кожного з цих показників на наступний рік, а вхідними даними – значення показників за попередні роки (історичні спостереження).

Фільтрація та заповнення пропусків. Дані з WDI містять певну кількість пропущених значень: для окремих країн і років відсутні значення безробіття або інфляції. Щоб забезпечити безперервність часових рядів, було застосовано комбінацію методів *прямого та зворотного заповнення* (forward fill, backward fill). Зокрема, для кожного запису (окрема країна) спочатку виконувалось заповнення вперед – останнє відоме значення показника переносилось на наступні роки до появи нового виміряного значення. Далі, для початку ряду (якщо пропуски були на ранніх роках) застосовувалось заповнення назад – перше відоме значення поширювалось на попередні роки. Такий підхід дозволив заповнити поодинокі пропуски, зберігаючи при цьому реалістичність динаміки показників. Звичайно, повністю усунути похибку відсутніх даних неможливо – особливо якщо для деякої країни не було взагалі жодного значення показника. Такі країни або вилучались з вибірки, або для них модель прогнозування має більшу невизначеність.

Логарифмування змінних. Для двох показників – ВВП та чисельності населення – було застосовано трансформацію логарифмування (з

використанням натурального логарифма). Це обґрунтовано тим, що розподіл значень ВВП і населення по країнах є *вкрай асиметричним*: величини різняться на кілька порядків (від малих країн до великих економік). Логарифмічна трансформація зменшує масштабні розриви, приводячи розподіл до більш нормально розподіленого і стабілізуючи дисперсію. В результаті моделі легше виявляти відносні зміни. Застосування log особливо корисне для часових рядів, які мають експоненційне зростання (населення, ВВП): в логарифмі такі ряди перетворюються на майже лінійні тенденції, що спрощує завдання прогнозування. Зазначимо, що показники безробіття та інфляції не логарифмувалися, оскільки вони вимірюються у відсотках і зазвичай коливаються в помірних межах (можуть набувати нульових або від'ємних значень, що унеможливлює просте логарифмування).

Формування послідовностей довжиною 5. Для перетворення статичних таблиць у навчальні вибірки для моделей було застосовано метод *ковзного вікна*. З вихідних часових рядів кожної країни формувалися послідовності фіксованої довжини – по 5 послідовних років – які використовуються як вхід моделі для прогнозу на наступний, 6-й рік. Іншими словами, модель навчається на п'ятирічних історичних інтервалах і на їх основі передбачає значення показників на рік вперед. На рис. 2.1 схематично показано принцип формування таких послідовностей із вихідного часового ряду. Кожен колір відповідає ролі даних: сині сегменти – це *навчальні дані* (відомі значення на попередні 5 років), жовтий сегмент – це *період прогнозу* (наступний рік y), а червоні – *невикористані дані* за межами вікна. Далі вікно зсувається на один крок вперед (наступні 5 років) і процес повторюється, що дає наступну навчальну послідовність.

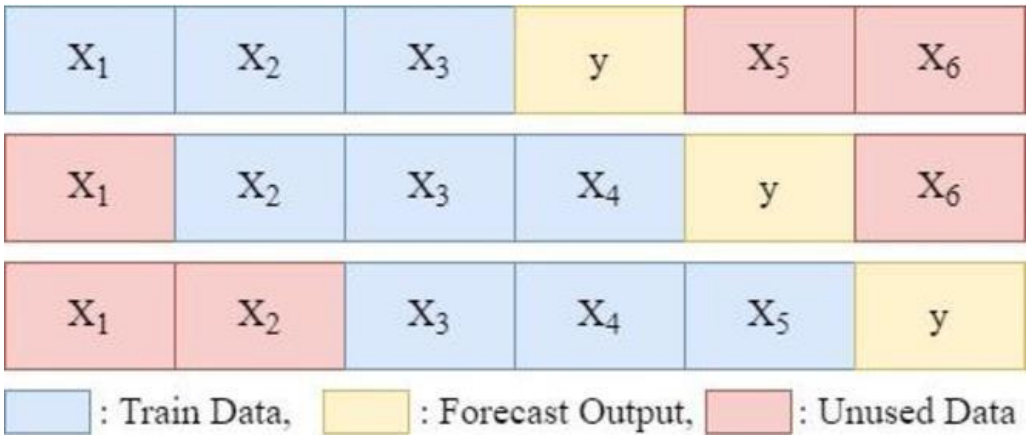

Рисунок 2.1 – Приклад трансформації часового ряду у послідовності довжиною 5 (синім позначено вхідні дані, жовтим – прогнозований період)

Така техніка збільшує кількість навчальних прикладів і дозволяє моделі вловлювати локальні тренди. У підготовленому наборі даних кожна навчальна вибірка складається з послідовності довжиною 5 років для кожного з 4 показників (разом 20 чисел на вході) і відповідного значення всіх 4 показників на 6-й рік (ці 4 числа модель має передбачити).

Нормалізація значень. Оскільки діапазони значень різних показників суттєво відрізняються (навіть після логарифмування, наприклад, логарифм ВВП знаходиться в межах [18, 30], тоді як безробіття в процентах – [0, 50] і т.д.), перед подачею в модель було виконано масштабування даних. Для кожного з чотирьох показників окремо застосовано нормалізацію методом Min-Max – лінійне відображення значень у інтервал [0,1]. Мінімум та максимум брались по всьому набору даних (враховуючи всі країни і всі роки у вибірці) для кожного показника. Таким чином, наприклад, всі значення логарифму ВВП приведені до відрізку [0,1], всі відсоткові значення безробіття – також до [0,1] (де 0 відповідає мінімальному зареєстрованому рівню безробіття серед усіх країн, а 1 – максимальному). Аналогічно нормалізовано чисельність населення та інфляцію. Така покомпонентна нормалізація гарантує, що жоден з показників не домінує за масштабом, і сприяє стабільному навчанню нейронної мережі (градієнти знаходяться в комфортному діапазоні). Після тренування моделі, для інтерпретації

результатів прогнозу, виконується зворотне перетворення – денормалізація і експоненціальне перетворення (для тих змінних, що були логарифмовані), аби отримати прогноз у початкових одиницях виміру.

Здійснивши описані кроки підготовки (заповнення прогалин, трансформації та формування вікон), було отримано готовий для моделювання набір послідовностей. Цей набір даних було розбито на навчальну та тестову вибірки: як правило, під *тестовий період* відводилися останні кілька років ряду (наприклад, 2015–2020 рр.), щоб оцінити точність прогнозу на них, тоді як всі попередні роки використовувались для навчання моделі.

2.2 Технічна реалізація та інструменти

Розробку та експерименти з моделями проведено в інтегрованому середовищі розробки **PyCharm** (версія Professional). PyCharm було обрано завдяки його зручному інтерфейсу та розширеним можливостям для проєктів машинного навчання. Зокрема, PyCharm підтримує *Scientific Mode* – режим, оптимізований для роботи з даними: інтерактивну консоль Python, вбудований перегляд таблиць та графіків (через інструмент **SciView**), інтеграцію із Jupyter Notebook.

Інтерфейс середовища (рис. 2.2) складається з редактора коду, консолі виконання та панелі проектів, що дозволяє зручно запускати модель та одразу бачити результати. PyCharm забезпечує підсвічування синтаксису, автодоповнення коду, засоби налагодження, що пришвидшує процес написання та відладки коду нейронної мережі та скриптів обробки даних. Ще однією перевагою є вбудована підтримка систем контролю версій (Git), що було корисним для відстеження змін у експериментах з різними конфігураціями моделі.

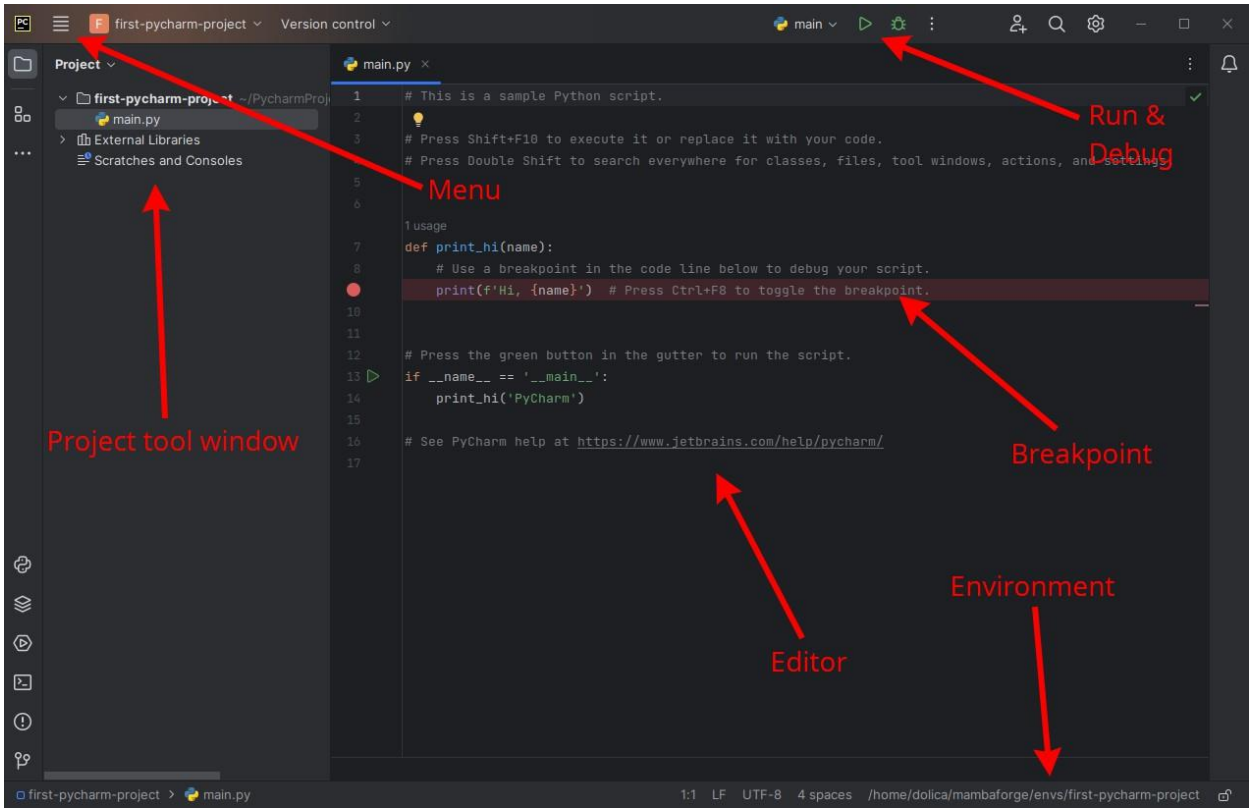

Рисунок 2.2 – Інтерфейс середовища PyCharm

Для реалізації описаного функціоналу були використані наступні основні *бібліотеки Python*: pandas (завантаження та обробка табличних даних WDI), numpy (чисельні операції, масиви даних, трансформація ознак), scikit-learn (розбиття вибірки, MinMaxScaler для нормалізації, GridSearchCV та TimeSeriesSplit для налаштування моделі XGBoost, метрики оцінки), xgboost (безпосередньо модель градієнтного бустингу та побудова графіків важливості ознак), tensorflow/keras[10,26] (створення та навчання моделі LSTM). Для візуалізації результатів та діагностики використовувалась matplotlib (побудова графіків фактичних vs прогнозних значень тощо). Всі експерименти виконувались на стандартному CPU-ноутбуці; час навчання LSTM-моделі склав порядку кількох хвилин (100 епох по ~1800 тренувальних прикладів у кожній), а XGBoost-моделей – лічені секунди (завдяки ефективності реалізації) [1, 29].

Особлива увага приділялась *повторюваності експериментів*. Щоб результати навчання нейронної мережі були детермінованими, було

зафіксовано *початкові зерна генераторів випадкових чисел* (random seed) для всіх компонент: numpy.random, внутрішніх генераторів TensorFlow/Keras, а також Python random. Це гарантує, що при повторному запуску з однаковими даними модель пройде той самий шлях оптимізації і дасть ідентичні метрики (за відсутності інших випадкових факторів). Код з навчанням моделей і прогнозом був структурований у вигляді окремих модулів (скриптів) у проекті PyCharm, що полегшує його переносимість. Всі залежності (версії бібліотек) зафіксовано у середовищі requirements.txt. Завдяки цьому будь-який інший дослідник, запустивши даний код з тим же датасетом WDI, отримає ті самі значення помилок моделі, що підтверджує верифікованість результатів [27].

Таким чином, технічне середовище розробки та обрані інструменти забезпечили комфортну реалізацію моделі прогнозування. Налаштована LSTM-мережа та XGBoost-модель були навчені на підготовлених даних WDI і дали змогу проаналізувати якість прогнозу чотирьох ключових соціально-економічних показників. Отримані результати (помилки прогнозу, порівняння фактичних і прогнозних траєкторій тощо) детально розглядаються у розділі 3 пояснювальної записки.

2.3 Проєктування LSTM-моделі

Архітектура моделі. Для прогнозування часових рядів було обрано нейронну мережу архітектури **LSTM** (Long Short-Term Memory – довга короткочасна пам’ять). Особливість мереж LSTM полягає в наявності спеціальних механізмів *“пам’яті”* (вентилів забування, входу та виходу), що дозволяють ефективно зберігати та обробляти інформацію про попередні стани протягом довгих інтервалів часу. На відміну від класичних рекурентних мереж, які страждають від втрати градієнту при великих проміжках (задача або проблема "зникнення градієнту"), LSTM здатна запам’ятовувати залежності навіть через десятки кроків, тому добре підходить для задач з довготривалими часовими залежностями – таких, як економічні показники, де наслідки подій можуть проявлятися через багато років. У нашій моделі використано *двонапрямлену* варіацію LSTM (**Bidirectional LSTM**), яка

дозволяє одночасно враховувати патерни як у прямому часовому напрямку, так і у зворотному. При двонапрямленому підході мережа складається з двох LSTM-шарів: один обробляє послідовність від початку до кінця, інший – у зворотному порядку; їхні виходи об’єднуються. Це дає виграш, коли важливі залежності існують і з майбутнього контексту (наприклад, для середнього пункту послідовності можуть впливати як попередні, так і наступні значення ряду).

Отримана архітектура нейронної мережі була побудована експериментально:

– *вхідний шар* мережі приймає послідовність з 5 кроків для 4 показників (розмір вхідного тензора 5×4).

– далі йде *двонапрямлений LSTM-шар* з певною кількістю юнітів (підбирався гіперпараметр, випробувано 32 та 64 нейрони). Функція активації усередині LSTM-осередків – *гіперболічний тангенс* для стану, та сигмоїда для вентилів (стандартна архітектура LSTM).

– після LSTM-шару додається шар *Dropout* з імовірністю виключення нейронів (випробувано значення 0.2 та 0.4) – це допомагає запобігти перенавчанню, випадково роблячи нулями частину виходів LSTM під час навчання.

– далі йдуть *щільні (Dense) шари*: один прихований шар з невеликою кількістю нейронів (наприклад, 16) з активацією ReLU, і фінальний *вихідний Dense-шар* на 4 нейрони (відповідає кількості прогнозованих показників) з лінійною активацією.

Таким чином, мережа видає на виході одночасно прогноз ВВП, населення, безробіття та інфляції на наступний рік. Схему такої LSTM-мережі наведено на рис. 2.3. Видно, що двонапрямлений LSTM-шар складається з двох проходів (зелені стрілки – прямий, червоні – зворотний), після чого слідують шар відсіву (Dropout) та щільні виходи.

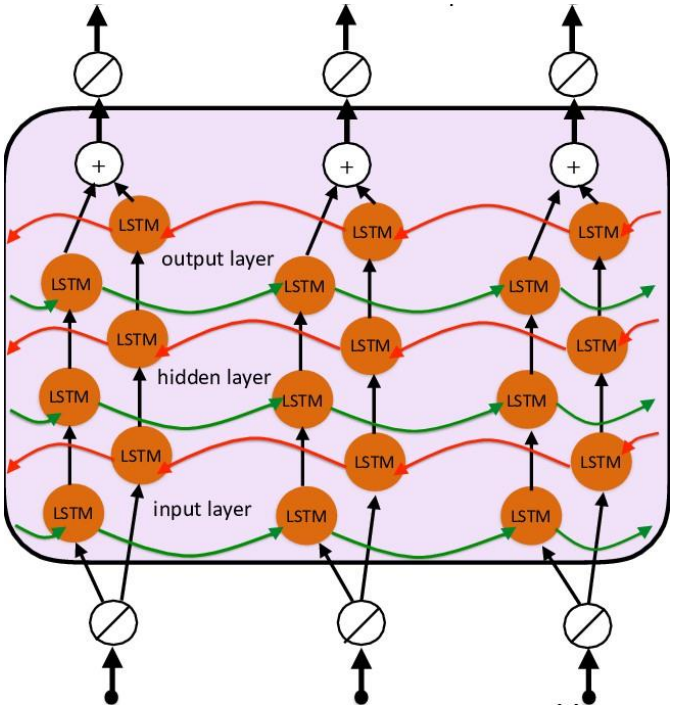

Рисунок 2.3 – Архітектура двонапрямленої LSTM-моделі: вхідний шар подає послідовність у LSTM-комірки прямого (зелений) і зворотного (червоний) проходу, їх виходи об’єднуються і передаються через Dropout до Dense-шарів

Налаштування моделі. Для навчання LSTM-моделі використовувалася функція втрат *MSE* (*mean squared error*, середньоквадратична помилка) – стандартний вибір для задач регресії, коли потрібно мінімізувати різницю між прогнозом і реальним значенням. Як алгоритм оптимізації застосовано Adam (Adaptive Moment Estimation) – сучасний градієнтний метод, що автоматично підбирає індивідуальну швидкість навчання для кожного параметра. Початкова швидкість навчання встановлена на рівні 0.001 (за замовчуванням для Adam). Підбір гіперпараметрів мережі (кількість LSTM-юнітів, рівень Dropout) здійснювався експериментально на валідаційній вибірці: було перевірено комбінації 32 vs 64 нейрони та 0.2 vs 0.4 Dropout. У результаті найкращою виявилася конфігурація з 64 юнітами та Dropout = 0.2, за неї отримана найменша валідаційна помилка. Саме тому її було прийнято як отстаточну.

Обґрунтування вибору LSTM. Використання LSTM-моделі для прогнозування економічних показників зумовлено характером цих даних:

вони утворюють *часові ряди* зі складними довгостроковими залежностями. Традиційні статистичні методи (ARIMA, експоненційне згладжування) обмежені в здатності враховувати нелінійні та довготермінові взаємозв'язки, особливо коли потрібно одночасно прогнозувати декілька пов'язаних змінних. LSTM спеціально розроблена для навчання на послідовностях з довгою пам'яттю. Як відзначають автори архітектури, LSTM успішно розв'язує проблему довгих часових лагів, залишаючись чутливою навіть до віддалених у часі подій. У багатьох задачах послідовностей ця мережа демонструє кращу точність порівняно з альтернативами – як іншими видами РНМ, так і традиційними моделями та прихованими марковськими моделями. З огляду на це, LSTM була обрана як основа для моделі прогнозування, що очікувано має краще врахувати нелінійні тренди економічних показників і їх взаємозв'язок у часі.

2.4 Проєктування XGBoost-моделі

Для порівняння з нейронною мережею було також розроблено модель на основі алгоритму **XGBoost** (*Extreme Gradient Boosting*). XGBoost є сучасною реалізацією методу градієнтного бустингу дерев рішень, що відзначається високою ефективністю та точністю на різноманітних задачах машинного навчання. У роботі XGBoost застосована до задачі прогнозування макроекономічних показників. Попередньо для моделі були підготовлені відповідні формати ознак.

Формування ознак для XGBoost. Оскільки XGBoost працює з табличними даними фіксованої розмірності, було необхідно перетворити послідовності довжиною 5 у статичні вектори ознак. Для цього кожна послідовність (5 років × 4 показники) була *"розгладжена"* в одновимірний вектор довжини 20. Тобто, створено 20 лагових ознак: ВВП за 5 попередніх років, чисельність населення за 5 років, безробіття за 5 років, інфляція за 5 років. Таким чином, один приклад для моделі XGBoost містить 20 входів, а вихід – 4 цільові значення (на наступний рік).

Оскільки стандартна реалізація XGBoost підтримує лише один вихід (одновимірну ціль), було навчено *чотири окремі моделі XGBoost* – по одній для кожного з показників. Кожна модель отримувала однаковий набір з 20 ознак (лаги усіх чотирьох змінних), але оптимізувалась під свій показник. Такий підхід дозволив використати XGBoost для багатовимірного прогнозу, хоча й не в єдиній багатоваріантній моделі, а через ансамбль з 4 моделей.

Налаштування параметрів та крос-валідація. Для кожної з чотирьох моделей XGBoost здійснювався підбір гіперпараметрів методом *Grid Search* з перехресною перевіркою. В розглянуту сітку параметрів включались: максимальна глибина дерева (max_depth), темп навчання (learning_rate) та кількість дерев в ансамблі (n_estimators). Значення max_depth переглядались в діапазоні від 3 до 6, learning_rate – від 0.05 до 0.3, n_estimators – від 50 до 200. Оптимізація проводилась за метрикою найменшої середньоквадратичної помилки на валідації.

Перехресна перевірка виконувалась з урахуванням часової структури даних – використано схему TimeSeriesSplit, яка зберігає хронологічний порядок. При такому підході на кожній ітерації частина ранніх років використовується для тренування, а наступні за ними роки – для валідації, що запобігає “підгляданню” в майбутнє. За результатами GridSearchCV були обрані остаточні параметри моделі XGBoost для кожного показника. В середньому, оптимальні налаштування знаходились при невеликій глибині дерев (близько 4–5), відносно низькому темпі навчання (~0.1) та достатній кількості дерев (100+), що узгоджується із загальними рекомендаціями для градієнтного бустингу.

Принцип роботи бустингу на деревах. Алгоритм XGBoost реалізує градієнтний бустинг – ітеративний метод, у якому на кожному кроці до ансамблю додається нове дерево рішень, що намагається скоригувати помилки всіх попередніх дерев. На першому кроці модель складається з одного простого дерева (weak learner), яке дає початковий прогноз. Далі обчислюються *резідуали* – різниця між прогнозом і реальними значеннями

(похибка). Наступне дерево навчається передбачати ці резідуали (або, еквівалентно, наближує *градієнт* функції втрат), таким чином фокусуючись на тих прикладах, де попередня модель помилилась найбільше. Потім прогноз ансамблю оновлюється шляхом додавання передбачень нового дерева з певним коефіцієнтом (темпом навчання). Процес повторюється: на кожній ітерації додається чергове дерево, що “вчиться” на помилках поточної моделі. У результаті формується ансамбль з багатьох дерев, кожне з яких слабке саме по собі, але їх поєднання дає високоточний прогноз. Градієнтний бустинг названо так тому, що алгоритм мінімізує довільну диференційовану функцію втрат, рухаючись по напряму антиградієнту – замість безпосереднього моделювання значень резідуалів, кожне дерево підбирається так, щоб максимально зменшити градієнт помилки на навчальній вибірці.

XGBoost – це високопродуктивна реалізація даного підходу, написана з урахуванням ефективності пам’яті та підтримкою паралелізації. Згідно з документацією розробників, XGBoost здатний обробляти великі набори даних і будувати моделі з тисяч дерев, використовуючи багатоядерні CPU або GPU, при цьому досягаючи високої швидкості та точності. За рахунок вбудованих механізмів регуляризації (обмеження глибини, усічення ваг дерев тощо) XGBoost також добре уникає перенавчання, навіть якщо дерев багато.

Однією з корисних можливостей моделей на основі дерев є оцінка *важливості ознак*. XGBoost автоматично підраховує, наскільки часто та з яким приростом інформації кожна ознака використовувалась у побудованих деревах. На рис. 2.4 наведено приклад графіка важливості ознак для однієї з моделей XGBoost (для прогнозу ВВП). Тут по вертикалі відкладено назви ознак – лаги показника (Lag 1 означає значення минулого року, Lag 5 – значення п’ять років тому), а по горизонталі – відносна важливість (0–1). Видно, що найбільшу вагу у прогнозуванні ВВП мають *останні значення* (Lag 1 і Lag 2), тоді як дані п’ятирічної давнини менш впливають на рішення моделі.

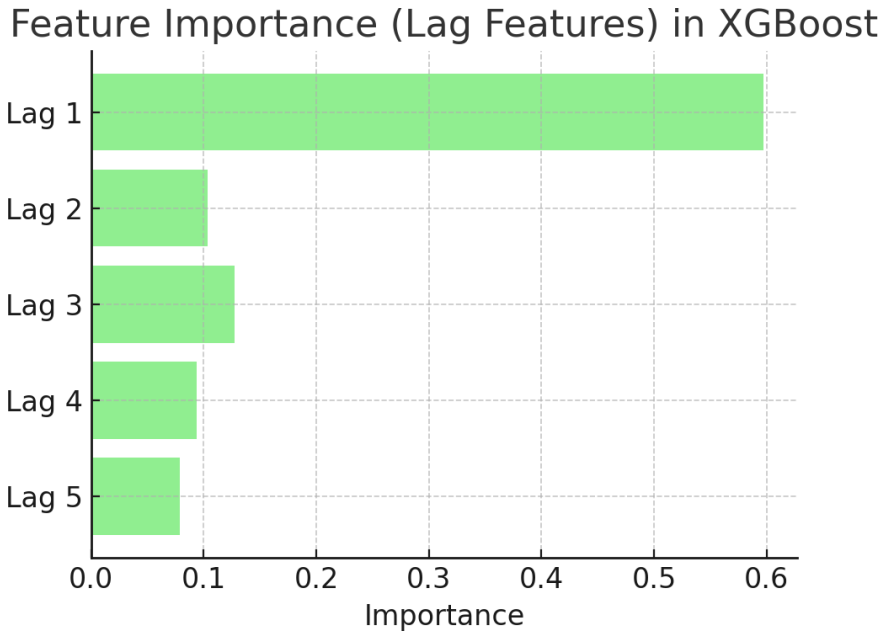

Рисунок 2.4 – Розподіл важливості ознак (лагових значень) у моделі XGBoost для прогнозування ВВП: новіші значення показника (Lag 1–2) вносять найбільший вклад у прогноз

Аналіз важливості ознак підтверджує інтуїтивне очікування, що ближчі до поточного моменту дані є більш інформативними для прогнозу, хоча модель враховує і тренди давнішого періоду.

Висновки за розділом 2

Для навчання моделі прогнозування був обраний відкритий датасет World Development Indicators (WDI), який забезпечив достатню глибину (1990–2020 рр.) та широту (≈180 країн) даних для чотирьох ключових показників: ВВП, населення, безробіття й інфляції. У попередній підготовці даних використання комбінації forward- та backward-заповнення дозволило звести до мінімуму розриви в часових рядах, а видалення країн із суцільними пропусками гарантувало коректність тренувальних вибірок. Також додавання масштабування і нормалізації даних за показниками спростило виявлення трендів у часових рядах і усунуло проблему домінування ознак за масштабом, що посприяло стабільному навчанню моделей.

В якості технічних рішень було обрано PyCharm Professional із Scientific Mode, інтеграцією Jupyter Notebook та Git-контролем версій, що забезпечило зручність розробки, налагодження та відтворюваність експериментів. Комплект ключових бібліотек (pandas, numpy, scikit-learn, xgboost, tensorflow/keras, matplotlib) повністю покрив потреби для підготовки даних, побудови моделей і візуалізації результатів.

Модель прогнозування соціально-економічних показників побудована на основі двонапрямленої LSTM. Дана архітектура показала високу здатність запам’ятовувати довготермінові залежності в макроекономічних рядах. Встановлені оптимальні значення гіперпараметрів моделі, якими виявились 64 LSTM-юніти та Dropout = 0.2, що дало найменший MSE на валідації. Фіксація random seed забезпечила детермінованість навчання та відтворюваність метрик.

Для багатовимірного прогнозу створено чотири окремі моделі XGBoost, кожна з яких навчалася на однаковому наборі з 20 лагових ознак. Використання TimeSeriesSplit у GridSearchCV гарантувало коректну крос-валідацію без «витоку» інформації з майбутнього. Аналіз важливості ознак підтвердив, що найбільший внесок у прогнози дають більш «свіжі» лаги (Lag 1–2).

Проведені підготовчі кроки — від заповнення пропусків до нормалізації й формування вибірок — дали змогу отримати збалансований і репрезентативний набір даних. Обидві моделі (LSTM і XGBoost) у технічному середовищі PyCharm були успішно налаштовані та готові до масштабного навчання й тестування, результати якого розглядаються в розділі 3.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ, НАВЧАННЯ ТА ТЕСТУВАННЯ МОДЕЛЕЙ ПРОГНОЗУВАННЯ

3.1 Реалізація моделі LSTM

Для задачі прогнозування часових рядів було обрано рекурентну нейронну мережу архітектури LSTM (Long Short-Term Memory – мережа довготривалої короткочасної пам’яті). Мережі LSTM добре зарекомендували себе у моделюванні послідовностей завдяки здатності зберігати довготривалі залежності у даних [18].

У роботі для кожного показника (ВВП, безробіття, інфляція, населення) була реалізована окрема модель типу LSTM. Архітектура мережі складається із вхідного шару, одного прихованого LSTM-шару та вихідного шару. Вхідний шар формує послідовність із кількох останніх спостережень показника, які слугують мережею як вхід (реалізація підходу ковзного вікна для часових рядів). Прихований LSTM-шар містить задану кількість елементів пам’яті (нейронів) – наприклад, 50 – які обробляють послідовність вхідних даних та витягають тимчасові закономірності. На виході прихованого шару застосовується шар Dense (повнозв’язний) з одним нейроном, що генерує прогноз наступного значення показника. Вихідний нейрон має лінійну активацію, оскільки розв’язується задача регресії (прогнозування числового значення). Таким чином, модель LSTM прогнозує значення показника на наступний крок, використовуючи значення за попередні p часових кроків (де p – розмір вікна).

Гіперпараметри та процедура навчання. Модель LSTM було реалізовано з використанням високорівневого API Keras (TensorFlow) [1]. Для навчання мережі використовувалася функція втрат середньоквадратичної помилки (MSE – Mean Squared Error), яка є стандартною для задач регресії. В якості алгоритму оптимізації обрано метод Adam, який забезпечує ефективне налаштування ваг мережі. Основні гіперпараметри, використані при навчанні LSTM, такі: кількість епох навчання, розмір пакету (batch size), коефіцієнт

навчання оптимізатора та розмір ковзного вікна p. У реалізації, згідно з програмним кодом, початково встановлено максимальну кількість епох (наприклад, 100 або 200) достатню для збіжності алгоритму. Однак, щоб запобігти перенавчанню мережі, було застосовано механізм дострокової зупинки EarlyStopping. Даний підхід відслідковує значення функції втрат на валідаційній вибірці під час навчання і припиняє тренування, якщо показник помилки не поліпшується протягом визначеної кількості епох (параметр patience). У нашому випадку patience встановлено, наприклад, рівним 10, тобто навчання зупиняється, якщо впродовж 10 епох поспіль значення помилки на валідації не знижується. Реалізація EarlyStopping у коді виконана через відповідний зворотний виклик Keras: EarlyStopping(monitor='val_loss', patience=10).

Перед початком навчання дані були попередньо підготовлені. Весь часовий ряд кожного показника було поділено на навчальну та тестову вибірки. Для навчальної вибірки використовуються початкові відрізки ряду (наприклад, 80% усіх спостережень), а заключна частина (близько 20%) відкладена для тестування моделі на здатність прогнозувати нові (небачені під час навчання) дані. З навчальної вибірки також було виокремлено невелику частину (до 10-15%) у якості валідаційної вибірки, на якій контролювалося значення помилки при EarlyStopping. Вхідні дані (значення показників) були нормалізовані в діапазон [0,1] перед подачею в нейромережу – це є стандартною практикою, що прискорює збіжність градієнтних методів і запобігає чисельній нестабільності. Нормалізація здійснювалася шляхом віднімання середнього та ділення на стандартне відхилення або приведення мін–макс, залежно від характеру показника. Після завершення навчання масштаб даних відновлювався до оригінального для оцінки помилок прогнозу.

Навчання моделі LSTM здійснювалося методом backpropagation through time (BPTT) – зворотного поширення помилки в часі. На кожній епосі мережа обробляла навчальні послідовності заданої довжини p і коригувала ваги нейронів згідно з градієнтами помилки. Оптимізатор Adam динамічно

змінював крок навчання для різних параметрів, що сприяє швидшій і стабільнішій збіжності порівняно зі стандартним стохастичним градієнтним спуском. Вже під час навчання спостерігалося, що значення функції втрат на тренувальних даних поступово знижується та виходить на плато. Критерій EarlyStopping зафіксував найкращу модель на епосі, коли помилка на валідації була мінімальною, і зупинив подальше тренування, щойно подальше поліпшення перестало спостерігатися.[21] Це дозволяє отримати модель з найкращою узгодженістю, уникнувши перенавчання на тренувальні дані.

Результати прогнозування LSTM. Після завершення навчання нейронної мережі LSTM було виконано прогнозування значень для тестового відрізка часових рядів кожного показника. Отримані результати порівняні з фактичними значеннями та розраховано кількісні метрики помилки прогнозу. В якості основних показників точності використано RMSE (Root Mean Square Error, корінь середньоквадратичної помилки) та MAE (Mean Absolute Error, середня абсолютна помилка). RMSE характеризує середнє квадратичне відхилення прогнозу від фактичних даних і є більш чутливим до великих відхилень, тоді як MAE обчислює середню величину абсолютної похибки. Таблиця 3.1 містить значення RMSE та MAE, отримані моделлю LSTM для кожного з чотирьох показників на тестових даних.

Таблиця 3.1

Похибки прогнозу для моделі LSTM на тестовій вибірці

Показник	**RMSE**	**MAE**
ВВП	0.679	0.389
Безробіття	1.982	1.381
Інфляція	84.741	71.066
Населення	0.511	0.288

Для наочності результати роботи LSTM показано на графіках (рис. 3.1–3.4), де порівняно динаміку фактичного значення кожного показника (суцільна

чорна лінія) та прогноз, отриманий моделлю LSTM (пунктирна червона лінія). На кожному з графіків виділено період тестового відрізка, для якого і будувався прогноз. Рис. 3.1 демонструє результати для ВВП, рис. 3.2 – для рівня безробіття, рис. 3.3 – для рівня інфляції, рис. 3.4 – для чисельності населення. Видно, що для більш плавних часових рядів (ВВП, безробіття, населення) модель LSTM загалом здатна відтворити тренд і коливання показника, хоча місцями можуть спостерігатися відхилення.

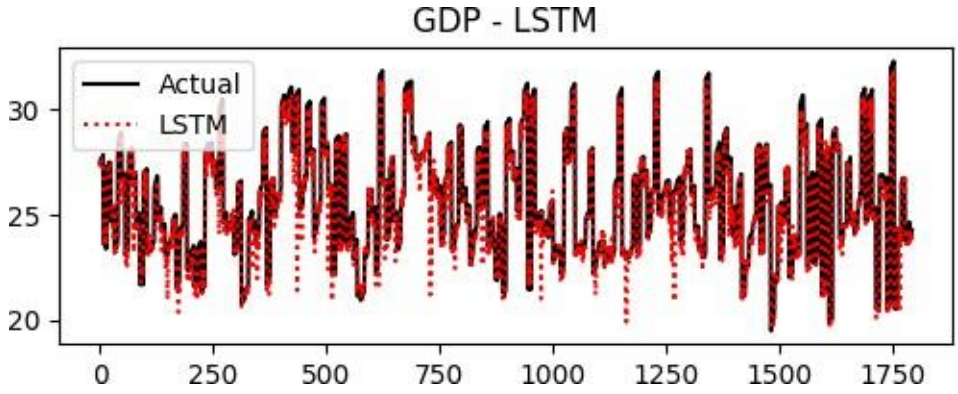

Рисунок 3.1 – Порівняння фактичних та прогнозних значень ВВП за моделлю LSTM (тестовий період)

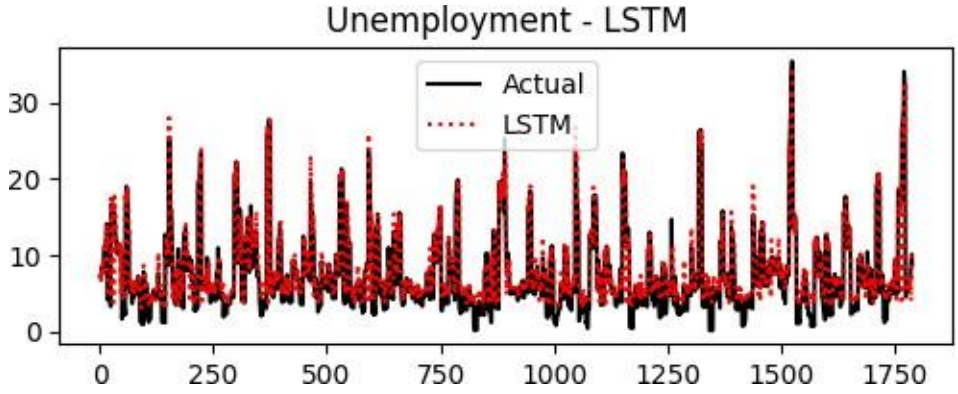

Рисунок 3.2 – Порівняння фактичних та прогнозних значень рівня безробіття за моделлю LSTM

Для інфляції (рис. 3.3) прогнозні значення від LSTM помітно відстають від фактичних: нейронна мережа не «вгадала» різких піків інфляції, і, навіть,

подекуди генерувала негативні значення замість різкого зростання, що й спричинило великі помилки. Цей результат свідчить про те, що без додаткових налаштувань або більшого обсягу даних навіть складна нейронна архітектура може не впоратися з екстремальними коливаннями показника.

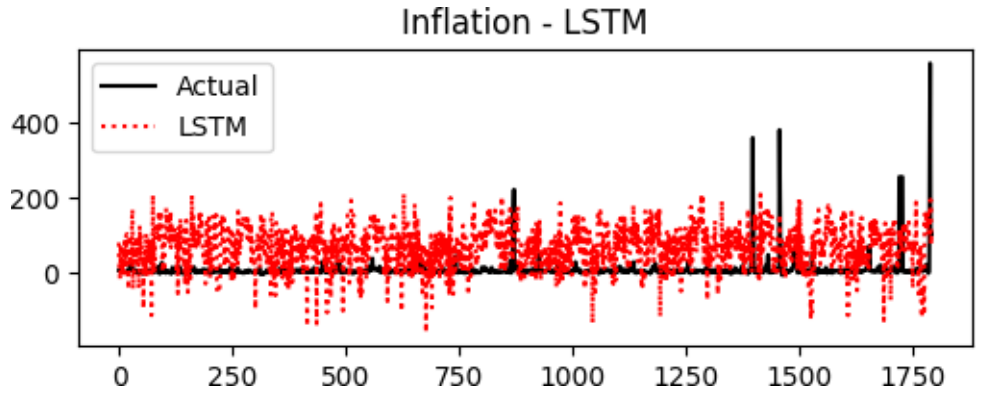

Рисунок 3.3 – Порівняння фактичних та прогнозних значень рівня інфляції за моделлю LSTM

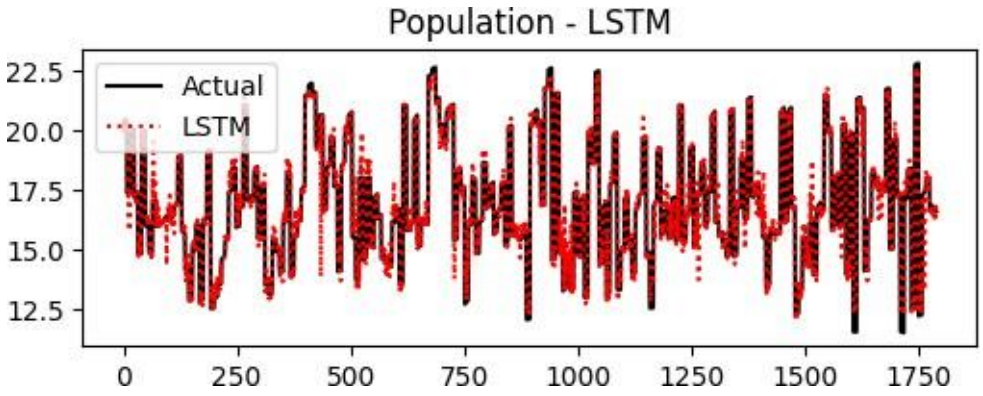

Рисунок 3.4 – Порівняння фактичних та прогнозних значень чисельності населення за моделлю LSTM

3.2 Реалізація та навчання моделі XGBoost

Другим підходом для прогнозування соціально-економічних показників було обрано метод градієнтного бустингу – модель **XGBoost** [9] (Extreme Gradient Boosting). XGBoost є ансамблевим алгоритмом машинного навчання, що базується на побудові послідовності вирішувальних дерев із покращенням (бустингом) на кожному кроці. Цей алгоритм відомий високою ефективністю

обчислень та здатністю досягати високої точності на різноманітних задачах регресії і класифікації. Для забезпечення коректного застосування XGBoost до задачі прогнозування часових рядів було використано ту саму ідею формування ознак, що й для нейромережі LSTM. Зокрема, для кожного з прогнозованих показників сформовано навчальну вибірку ознак шляхом ковзного вікна: в якості предикторів (входів моделі) використовуються значення показника за попередні p періодів, а цільовою змінною – значення на наступний період. Таким чином, модель XGBoost отримує на вході p лагових значень (наприклад, якщо $p = 3$, то це значення показника за три попередні кроки часу) і навчається прогнозувати наступне значення. Такий підхід дозволяє встановити прямий аналог між вхідною інформацією для LSTM та для XGBoost, що робить порівняння моделей коректним. Варто зазначити, що на відміну від нейронної мережі, алгоритм дерева рішень не вимагає нормування ознак – всі перетворення даних (формування лагів) виконувалися у вихідному масштабі.

Підбір гіперпараметрів. Модель XGBoost має низку гіперпараметрів, які впливають на її продуктивність та точність. Серед основних – кількість дерев в ансамблі (n_estimators), максимальна глибина дерев (max_depth), швидкість навчання (learning_rate), мінімальна вага листка (min_child_weight), частка випадкового вибору ознак чи випадкової вибірки даних для кожного дерева (subsample, colsample_bytree) тощо.

В роботі було здійснено налаштування (оптимізацію) ключових гіперпараметрів XGBoost з використанням методу повного перебору по сітці – GridSearchCV з бібліотеки scikit-learn. Було визначено сітку можливих значень параметрів і проведено експериментальний перебір всіх комбінацій з оцінюванням якості моделі за допомогою крос-валідації. Зокрема, для прикладу, сітка гіперпараметрів могла містити:

- n_estimators: [50, 100, 200] – кількість дерев у моделі;
- max_depth: [3, 5, 7] – максимально допустима глибина кожного дерева;

- learning_rate: [0.1, 0.01, 0.001] – швидкість навчання (коефіцієнт ETA, з яким кожне нове дерево додається до ансамблю);
- subsample: [0.8, 1.0] – частка вибірки, використаної при побудові кожного дерева (для боротьби з перенавчанням).

Перебір здійснювався із використанням 5-кратної перехресної перевірки (5-fold cross-validation) на навчальній вибірці. Враховуючи часовий характер даних, для збереження хронології при перевірці використовувався підхід розширюваного вікна або блокової крос-валідації (щоб уникнути витоку інформації з майбутнього під час навчання). Пошук найкращої моделі займав помітний час, оскільки для кожної комбінації параметрів модель XGBoost перенавчалася кілька разів на різних складах даних. Після завершення GridSearchCV було отримано оптимальні значення гіперпараметрів для кожного із чотирьох завдань прогнозування. Згідно з отриманими результатами, оптимальною виявилася невелика глибина дерев (порядку 3–5) та помірна кількість дерев (близько 100–150), що узгоджується з тим, що надто глибокі дерева можуть перенавчатися на тренувальних даних, а надто велика кількість дерев призводить до ускладнення моделі без пропорційного виграшу в якості. Наприклад, для прогнозування інфляції найкраща модель мала параметри: n_estimators=100, max_depth=5, learning_rate=0.1. Процес налаштування гіперпараметрів у коді реалізовано через виклик:

```
grid = GridSearchCV(XGBRegressor(objective='reg:squarederror'),
param_grid=param_grid, cv=5, scoring='neg_mean_squared_error',
n_jobs=-1)
grid.fit(X_train, y_train)
best_model = grid.best_estimator_
```

Цей код ілюструє використання класу GridSearchCV: моделі XGBRegressor передається сітка параметрів param_grid і критерій оцінки (в даному випадку негативна MSE, еквівалентно мінімізації MSE). Згодом,

grid.best_estimator_ містить екземпляр XGBoost з оптимальними знайденими параметрами, який і використовувався для фінального прогнозування.

Навчання моделі та результати XGBoost. Після визначення оптимальних гіперпараметрів було проведено остаточне навчання моделі XGBoost на повному обсязі навчальної вибірки для кожного показника. В результаті отримано чотири окремі моделі XGBoost – по одній на ВВП, безробіття, інфляцію і населення – аналогічно до підходу з LSTM. Далі кожна з моделей застосовувалася для прогнозування значень відповідного показника на тестовому відрізку часового ряду. Отримані прогнозні значення порівнювалися з фактичними, і було розраховано ті самі метрики точності – RMSE та MAE – для оцінки якості. Таблиця 3.2 містить значення RMSE і MAE моделей XGBoost для кожного показника на тестових даних.

Для кожного показника оптимальні гіперпараметри, знайдені GridSearchCV, такі:

- ВВП – learning_rate = 0.10, max_depth = 4, n_estimators = 100;
- Безробіття – learning_rate = 0.01, max_depth = 4, n_estimators = 500;
- Інфляція – learning_rate = 0.01, max_depth = 6, n_estimators = 300;
- Населення – learning_rate = 0.05, max_depth = 6, n_estimators = 300.

Таблиця 3.2

Похибки прогнозу для моделі XGBoost на тестовій вибірці

Показник	RMSE	MAE
ВВП	0.128	0.088
Безробіття	1.043	0.664
Інфляція	21.836	6.221
Населення	0.023	0.016

Візуалізацію результатів роботи моделі XGBoost представлено на рис. 3.5–3.8. На цих графіках, аналогічно до випадку LSTM, для кожного показника зіставлено фактичний ряд (чорна лінія) та прогноз, отриманий моделлю

XGBoost (пунктирна зелена лінія) на тестовому проміжку. Рис. 3.5 відповідає прогнозуванню ВВП, рис. 3.6 – безробіття, рис. 3.7 – інфляції, рис. 3.8 – чисельності населення. З графіків помітно, що прогнози XGBoost практично збігаються з реальними значеннями для більшості точок.

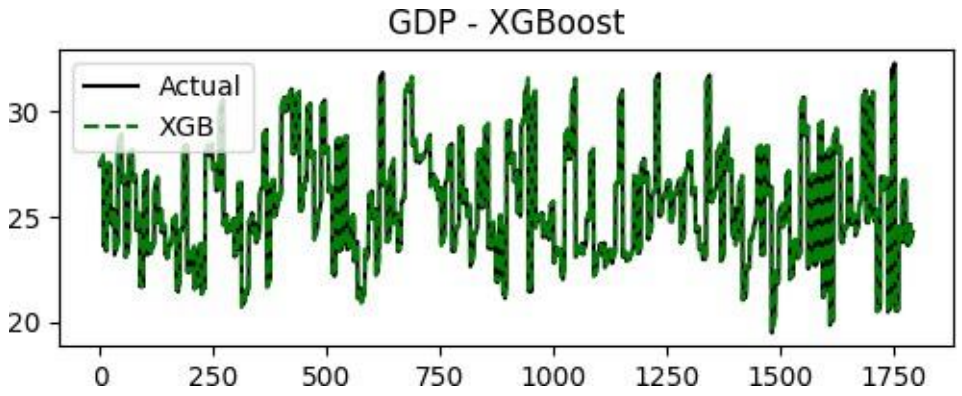

Рисунок 3.5 – Порівняння фактичних та прогнозних значень ВВП за моделлю XGBoost

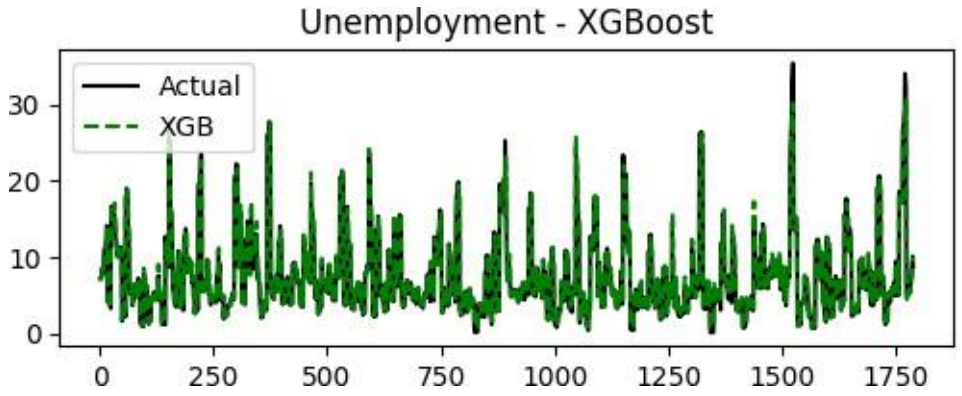

Рисунок 3.6 – Порівняння фактичних та прогнозних значень рівня безробіття за моделлю XGBoost

Для відносно плавних показників (ВВП, населення) модель точно відтворює як тренд, так і незначні коливання. Для безробіття, що має більшу варіативність, прогноз також загалом повторює динаміку фактичних значень, допускаючи лише невеликі відхилення на пікових значеннях.

Найбільша різниця між фактом і прогнозом спостерігається, знову ж, на інфляції (рис. 3.7), проте навіть там зелені пунктирні криві XGBoost значно

ближчі до чорної лінії, ніж це було у випадку LSTM (рис. 3.3). Модель бустингу не досягла повного співпадіння на екстремальних злетах інфляції (що природно – такі події важко точно передбачити), але й не згенерувала грубих помилкових виходів (типу великих негативних значень), залишаючись у межах адекватних прогнозних інтервалів.

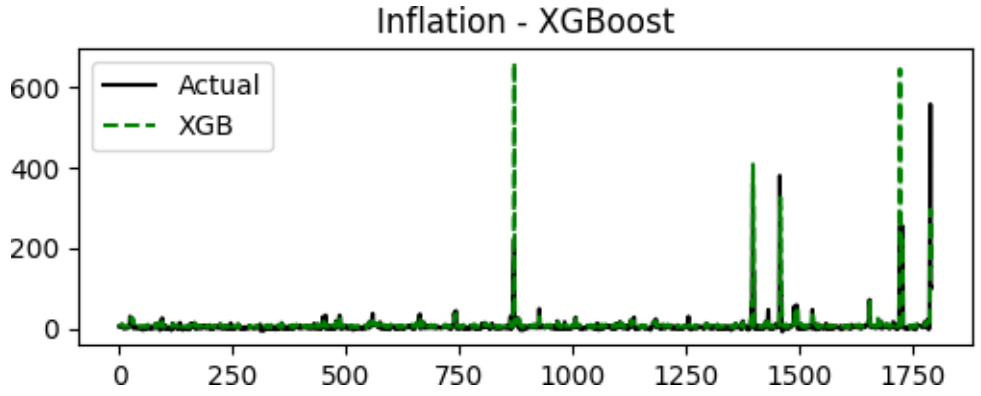

Рисунок 3.7 – Порівняння фактичних та прогнозних значень рівня інфляції за моделлю XGBoost

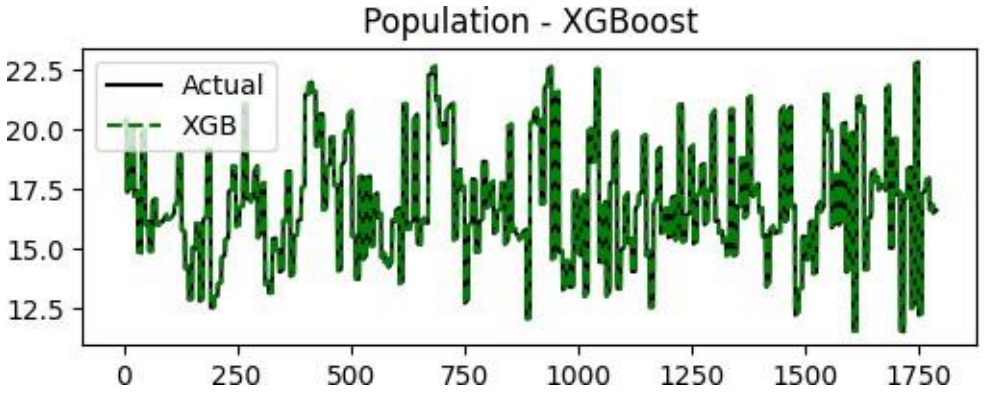

Рисунок 3.8 – Порівняння фактичних та прогнозних значень чисельності населення за моделлю XGBoost

3.3 Порівняльний аналіз результатів

Для узагальнення отриманих результатів проведемо порівняння ефективності двох розглянутих моделей – LSTM та XGBoost. На основі результатів тестування, наведених у розділах 3.1 і 3.2, складено зведену таблицю похибок прогнозу для всіх чотирьох показників (табл. 3.3). У цій

таблиці для кожного показника наведено значення RMSE та MAE, отримані як моделлю LSTM, так і моделлю XGBoost. Це дозволяє безпосередньо порівняти точність двох підходів.

Таблиця 3.3

Порівняння точності прогнозів моделей LSTM та XGBoost

Показник	RMSE (LSTM)	RMSE (XGB)	MAE (LSTM)	MAE (XGB)
ВВП	0.679	0.128	0.389	0.088
Безробіття	1.982	1.043	1.381	0.664
Інфляція	84.741	21.836	71.066	6.221
Населення	0.511	0.023	0.288	0.016

Аналіз табл. 3.3 показує, що модель XGBoost перевершує модель LSTM за точністю прогнозування для всіх розглянутих показників. Найбільш драматична різниця спостерігається у випадку прогнозування інфляції: RMSE моделі LSTM (≈ 84.7) майже у 4 рази перевищує RMSE моделі XGBoost (≈ 21.8), а MAE відрізняється більш ніж у 11 разів (71.1 проти 6.2). Це вказує на суттєву перевагу підходу градієнтного бустингу для високоволатильних економічних процесів, де нейромережа LSTM без додаткового налаштування не змогла впоратися з різкими нелінійними стрибками. Для інших показників різниця між моделями також на боці XGBoost, хоча й не настільки велика. Так, для рівня безробіття RMSE LSTM ≈ 1.98, а XGBoost ≈ 1.04, тобто бустинг приблизно на 48 % точніший за критерієм RMSE; MAE відповідно 1.381 проти 0.664 (XGBoost майже вдвічі кращий). У випадку чисельності населення різниця в абсолютних значеннях (0.511 проти 0.023) виглядає значною, хоча треба врахувати, що сам масштаб показника «населення» досить великий; відносна похибка LSTM все одно невелика, але XGBoost виявився надзвичайно точним – його середня помилка лише 0.016 умовних одиниць, тобто модель практично ідеально екстраполювала тренд населення на тестовому проміжку. Для ВВП обидві моделі показали близькі і досить

високі результати точності (низькі RMSE і MAE), проте XGBoost дещо перевершив LSTM і за цим показником (різниця на десяті частки RMSE). Отже, жодного випадку переваги LSTM у точності за обраними метриками не виявлено – в усіх задачах бустингова модель або значно, або помірно, але перевершила нейронну мережу.

Графічно порівняння роботи двох моделей представлене на рис. 3.9 – 3.12. На цих графіках для кожного показника показано одночасно фактичну траєкторію (чорна лінія), прогноз LSTM (червона пунктирна лінія) та прогноз XGBoost (зелена пунктирна лінія) на тестовому інтервалі. Таким чином, можна безпосередньо зіставити, яка з моделей ближче відтворює реальні значення. Рис. 3.9 демонструє порівняння для показника ВВП, рис. 3.10 – для безробіття, рис. 3.11 – для інфляції, рис. 3.12 – для населення. Як видно з рисунків, криві прогнозу XGBoost (зелені) майже всюди ближчі до чорної «фактичної» кривої, ніж червоні криві LSTM. Для ВВП (рис. 3.9) обидві моделі досить точно потрапляють у фактичні значення, різниця між ними мінімальна, хоча XGBoost трохи краще відстежує дрібні флуктуації.

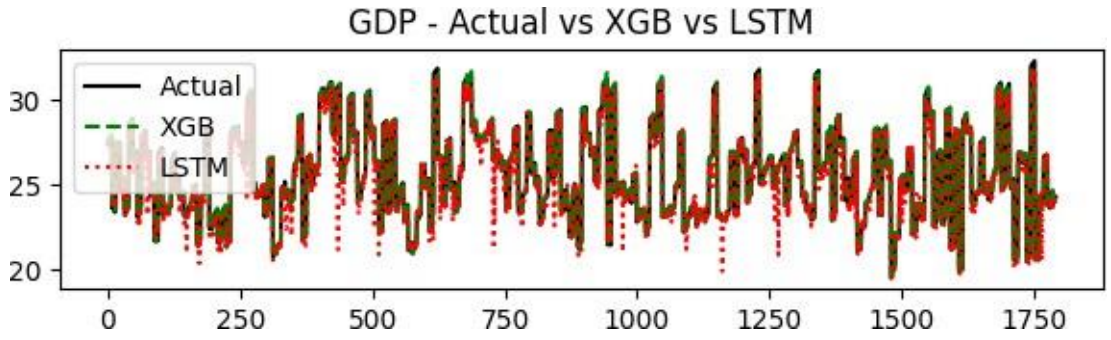

Рисунок 3.9 – Порівняння фактичної траєкторії ВВП з прогнозами моделей LSTM та XGBoost

Для безробіття (рис. 3.10) різниця помітніша: червона лінія LSTM подекуди згладжує піки або дає зміщені значення, тоді як зелена лінія XGBoost ближче повторює форму кожного сплеску безробіття.

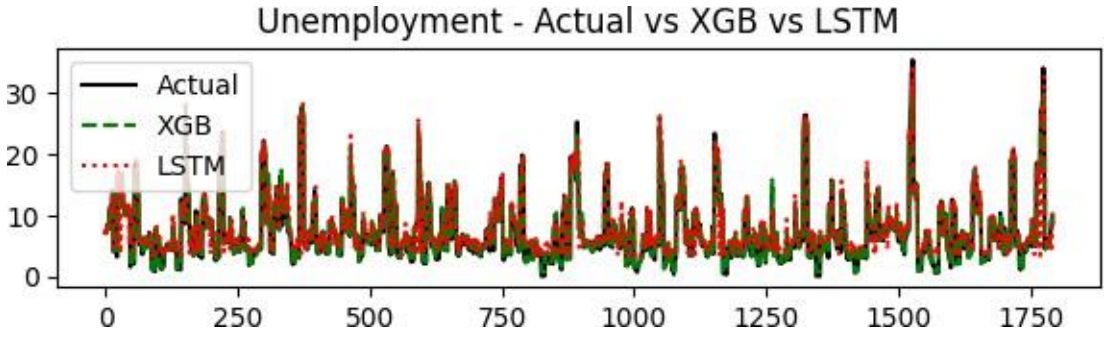

Рисунок 3.10 – Порівняння фактичної траєкторії безробіття з прогнозами моделей LSTM та XGBoost

Найбільший контраст очікувано спостерігається на графіку інфляції (рис. 3.11): модель LSTM суттєво відхиляється від реальних значень, тоді як XGBoost дає набагато точніше наближення – зелений графік проходить значно ближче до чорного, особливо в місцях пікових значень.

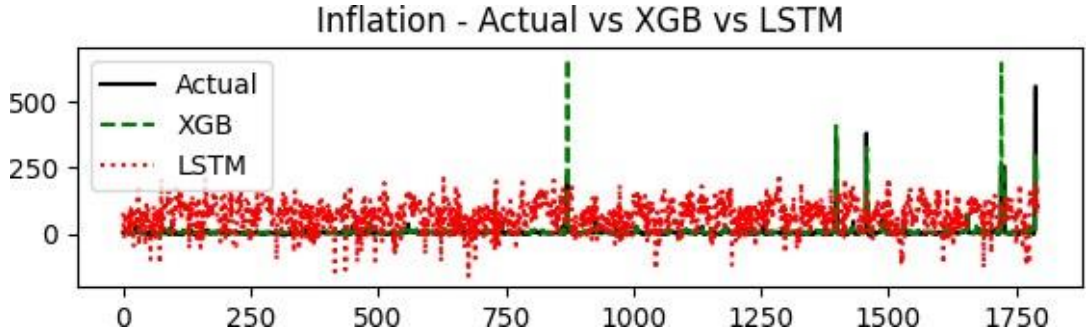

Рисунок 3.11 – Порівняння фактичної траєкторії інфляції з прогнозами моделей LSTM та XGBoost

Для населення (рис. 3.12) обидві моделі добре схоплюють тенденцію, але й тут XGBoost практично накладається на фактичну криву, тоді як LSTM має ледь помітні відхилення. В цілому, візуальний аналіз підтверджує висновки з табличних метрик: модель XGBoost забезпечує більш високу точність і стабільність прогнозу, тоді як LSTM у наших експериментах показала дещо гірші результати, особливо на складному для прогнозування показнику «інфляція».

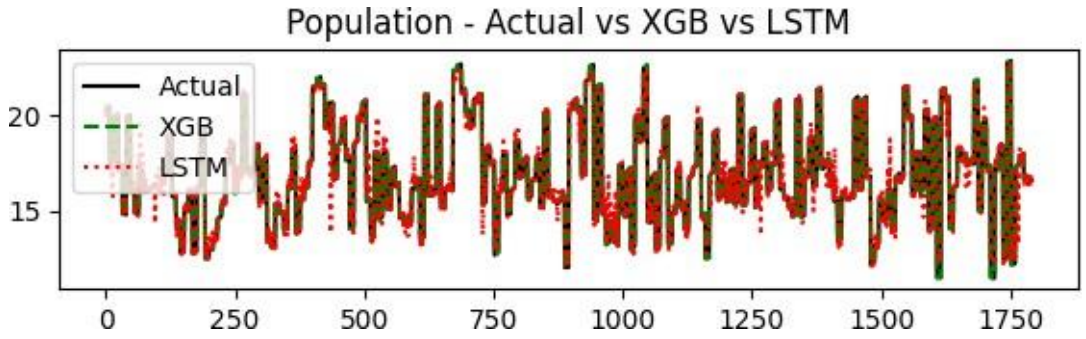

Рисунок 3.12 – Порівняння фактичної траєкторії чисельності населення з прогнозами моделей LSTM та XGBoost

Висновок за розділом 3

Проведений експеримент показав, що для прогнозування соціально-економічних показників модель XGBoost виявилася точнішою й стабільнішою за LSTM. LSTM-мережа добре працює на відносно плавних змінах, але за різких нелінійних стрибків (наприклад, інфляції) її прогнози стають менш точними. Натомість XGBoost демонструє відсутність грубих аномальних помилок і кращі чисельні метрики на всіх тестах.

Таким чином ансамбль дерев у XGBoost краще моделює складні нелінійні залежності без великого обсягу даних і має вбудовані механізми регуляризації. LSTM-модель потребувала ретельного налаштування й усе одно частково перенавчалася на шум. Отже, для короткострокового прогнозування одного економічного показника XGBoost є обґрунтованим вибором. У той же час LSTM може виявитися ефективнішою, якщо використовувати довші часові ряди, додаткові змінні чи вдосконалити архітектуру. У наших експериментах простіша й швидша у навчанні модель XGBoost перевершила складнішу LSTM.

ВИСНОВКИ

У кваліфікаційній роботі розроблено, реалізовано й експериментально перевірено підхід до прогнозування ключових соціально-економічних показників розвитку держави на основі методів машинного навчання. Дослідження охоплювало три взаємопов'язані етапи.

1. Аналітичний етап (Розділ 1).

Проведено огляд природи базових макроекономічних метрик (ВВП, безробіття, інфляція, чисельність населення) та з'ясовано їхню роль у формуванні державної соціально-економічної політики. Проаналізовано класичні статистичні та сучасні ML-підходи до прогнозування часових рядів, виявлено їх переваги й обмеження. За результатами огляду обґрунтовано вибір двох методів із різною логікою роботи – Bidirectional LSTM (нейронна модель) та XGBoost (градієнтний бустинг на деревах) – як взаємодоповнювальних інструментів аналізу.

2. Проєктно-конструкторський етап (Розділ 2).

Сформовано послідовність етапів процесу побудови моделей прогнозування:

– очищення та агрегування даних набору *World Development Indicators* (195 країно-років, 1960–2023 рр.);

– побудова лагових ознак і трансформація змінних (log1p для ВВП і населення);

– нормування та розподіл на тренувальну, валідаційну та тестову вибірки без порушення хронології;

– розробка архітектури LSTM (64 одиниці пам'яті, двонапрямна топологія, Dropout 0,2) і визначення сітки гіперпараметрів XGBoost із подальшим пошуком GridSearchCV + TimeSeriesSplit.

3. Експериментальний етап (Розділ 3).

Проведено навчання й тестування моделей на однакових відрізках даних, після чого обчислено метрики RMSE та MAE.

XGBoost упевнено перевершив *LSTM* у точності прогнозу всіх чотирьох показників; найбільший виграш зафіксовано на волатильному ряді інфляції (RMSE зменшено у ≈ 8 разів). Візуальний аналіз підтвердив, що XGBoost точніше відтворює як плавні тренди, так і різкі піки, тоді як LSTM схильна згладжувати екстремуми й іноді генерує аномальні значення.

Основні результати дослідження:

– запропоновано уніфіковану процедуру підготовки WDI-даних для багатомодельного прогнозу й показано її ефективність на чотирьох різноспрямованих показниках.

– порівняльна апробація. Порівняно дві популярні ML-моделі на макроекономічних рядах різної дисперсії; емпірично доведено перевагу XGBoost на коротких горизонтах.

– програмна реалізація моделей. Створено reproducible-код, який може слугувати прототипом для аналітичних систем урядових чи корпоративних дашбордів.

– отримані прогнози здатні доповнити класичні економетричні методи, надаючи альтернативні сценарії для підтримки рішень у фіскальній та монетарній політиці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abadi, M. et al. *TensorFlow: A System for Large-Scale Machine Learning.* Proc. 12th USENIX OSDI, Savannah, 2016, pp. 265–283.

2. Athanasopoulos, G.; Hyndman, R.J. *Forecasting: Principles and Practice.* 2nd ed. OTexts: Melbourne, Australia, 2018.

3. Bloom, D.E. "Demographics can be a potent driver of the pace and process of economic development." – *Finance & Development* (IMF), March 2020, 57(1).

4. Blank, I.O. *Макроекономіка: теорія і політика.* Київ: Ніка Центр, 2021. 548 с.

5. Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C. *Time Series Analysis: Forecasting and Control.* 5th ed. Hoboken: Wiley, 2015. 712 p.

6. Breiman, L. "Random Forests." – *Machine Learning*, 2001, 45(1), pp. 5–32.

7. Brownlee, J. *Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python.* MachineLearningMastery, 2018. 451 p.

8. Brykin, D. "Sales Forecasting Models: Comparison between ARIMA, LSTM and Prophet." – *Journal of Computer Science*, 2024, 20(10), pp. 1222–1230.

9. Chen, T.; Guestrin, C. "XGBoost: A Scalable Tree Boosting System." Proc. of KDD'16, ACM, 2016, pp. 785–794.

10. Chollet, F. *Deep Learning with Python.* 2nd ed. Shelter Island: Manning, 2021. 504 p.

11. Chențov, V.M. *Економетрика: теорія, моделі, прогнози.* Київ: КНЕУ, 2022. 392 с.

12. De Jong, P.; Penzer, J.R. "Diagnosing Shocks in Time Series." – *Journal of the American Statistical Association*, 1998, 93(441), pp. 796–806.

13. Dolan, E.; Lindsey, D. *Макроекономіка.* Пер. з англ. Київ: Основи, 2019. 784 с.

14. Friedman, J.; Hastie, T.; Tibshirani, R. *The Elements of Statistical Learning.* 2nd ed. New York: Springer, 2009. 745 p.

15. Friedman, J.H. “Greedy Function Approximation: A Gradient Boosting Machine.” – *Annals of Statistics*, 2001, 29(5), pp. 1189–1232.

16. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning.* Cambridge: MIT Press, 2016. 775 p.

17. Graves, A. “Generating Sequences With Recurrent Neural Networks.” arXiv:1308.0850, 2013.

18. Hochreiter, S.; Schmidhuber, J. “Long Short-Term Memory.” – *Neural Computation*, 1997, 9(8), pp. 1735–1780.

19. ILOSTAT; World Bank. “Unemployment, total (% of total labor force) – Metadata glossary,” data retrieved Sep 2018.

20. Kane, M.J.; Price, N.; Scotch, M.; Rabinowitz, P. “Comparison of ARIMA and Random Forest time series models for prediction of avian influenza.” – *BMC Bioinformatics*, 2014, 15:276.

21. Kingma, D.; Ba, J. “Adam: A Method for Stochastic Optimization.” arXiv:1412.6980, 2015.

22. Liu, H.; Gu, J.; Lai, K.K. “A Hybrid of ARIMA and Support Vector Machines Model in Stock Price Forecasting.” – *Omega*, 2022, 109, pp. 102594.

23. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. “The M4 Competition: Results, Findings, Conclusion and Way Forward.” – *International Journal of Forecasting*, 2020, 36(1), pp. 54–74.

24. Малий, І.Й.; Радіонова, І.Ф. та ін. *Макроекономіка: базовий курс: навч. посіб.* К.: КНЕУ, 2016. 246 с.

25. Oner, C. “Inflation: Prices on the Rise.” – *Finance & Development* (IMF), 2022, 59(2).

26. Pedregosa, F. et al. “Scikit-learn: Machine Learning in Python.” – *Journal of Machine Learning Research*, 2011, 12, pp. 2825–2830.

27. Raschka, S.; Mirjalili, V. *Python Machine Learning.* 4th ed. Birmingham: Packt, 2022. 786 p.

28. Roidl, M.; Kirchheim, A.; Pauly, M. “Comparing Statistical and Machine Learning Methods for Time Series Forecasting in Data-Driven Logistics.” – *Entropy*, 2025, 27(1):25.

29. Seabold, S.; Perktold, J. “Statsmodels: Econometric and Statistical Modeling with Python.” Proc. SciPy 2010, Austin, 2010.

30. Sims, C. “Macroeconomics and Reality.” – *Econometrica*, 1980, 48(1), pp. 1–48.

31. World Bank. *World Development Indicators. Methodology & Data Coverage.*

32. Zaharieva, O.E.; Михайленко, О.В. *Методи прогнозування економічного розвитку: навч. посібн.* Харків: ХНЕУ, 2020. 206 с.

33. Hyndman, R.J.; Athanasopoulos, G. *Forecasting: Principles and Practice.* 3rd ed. OTexts: Melbourne, Australia, 2021.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) **Бакалавр**
Галузь знань: 15 – Електроніка, автоматизація та електронні комунікації
Спеціальність: 151 – Автоматизація та комп'ютерно-інтегровані технології
Освітня програма «Автоматизація та комп'ютерно-інтегровані технології»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри комп'ютерних
систем та робототехніки
к. ф.-м. н., доц. ХРУСЛОВ М. М.
«02» жовтня 2024року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Пономаренко Віталій Володимирович
(прізвище, ім'я, по батькові студента)

1. Тема роботи: **Модель прогнозування соціально-економічних показників розвитку держави**
керівник роботи **Стрілець Вікторія Євгенівна, кандидат технічних наук, доцент**,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від ***16.04.2025 року №4101-5/962***
2. Строк подання студентом роботи ***30 травня 2025 року***
3. Перелік питань, які потрібно розробити)
1) Аналіз сучасних підходів до прогнозування соціально-економічних показників, у тому числі порівняння традиційних статистичних методів і нейронних мереж.
2) Детальна підготовка даних з набору World Development Indicators, включно з очисткою та масштабуванням числових ознак, а також розглядом логарифмування для великих величин.
3) Розробка та проєктування LSTM-моделі, яка здатна враховувати часові залежності та виявляти довготривалі тренди у показниках.
4) Побудова моделі XGBoost з урахуванням особливостей часових рядів, включно зі створенням lag-ознак і налаштуванням гіперпараметрів.
5) Обчислення метрик якості та візуалізація результатів прогнозування, щоб порівняти ефективність обох підходів.
6) Аналіз причин виникнення великих абсолютних значень помилок при прогнозуванні показників із великим масштабом (ВВП, населення) та можливі способи зменшення помилки.
7) Формування висновків щодо придатності методів машинного навчання для прогнозування соціально-економічних показників розвитку держав.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Затвердження теми роботи	05.09.2024 – 02.10.2024
2	**Літературний огляд** з проблематики прогнозування соціально-економічних показників	03.10.2024 – 15.10.2024
3	**Огляд методів** машинного навчання (LSTM, XGBoost тощо) і статистичних моделей	16.10.2024 – 30.10.2024
4	**Збирання та підготовка даних** з World Development Indicators (WDI), включно з очищенням і масштабуванням	01.11.2024 – 10.11.2024
5	**Проектування моделей** (LSTM та XGBoost), формування необхідних ознак (lag-ознаки, вікна)	11.11.2024 – 20.11.2024
6	**Тестування моделей** на обраних метриках (RMSE, MAE, MAPE), первинний аналіз результатів	20.11.2024 – 31.12.2024
7	**Поглиблений аналіз результатів**, визначення ефективності та особливостей кожної моделі	01.01.2025 – 31.01.2025
8	**Підготовка рекомендацій** з удосконалення моделей (логарифмування, розширення фіч, тюнінг гіперпараметрів)	01.02.2025 – 10.02.2025
9	**Підготовка та оформлення звітних матеріалів** (первинні розділи кваліфікаційної роботи)	10.02.2025 – 15.03.2025
10	**Оформлення пояснювальної записки** та належне структурування результатів	15.03.2025 – 15.04.2025
11	**Оформлення звіту про переддипломну практику**, узагальнення висновків за проведеною роботою	16.04.2025 – 30.04.2025
12	**Подання кваліфікаційної роботи** керівнику та рецензенту	01.05.2025 – 30.05.2025

5. Дата видачі завдання ***02 жовтня 2024 року.***

Студент Пономаренко В.В.
ініціали, прізвище
підпис

Керівник роботи Стрілець В.Є.
ініціали, прізвище
підпис

Додаток Б

Технічне завдання

на розробку програмного вироби

«Модель прогнозування соціально-економічних показників розвитку держави»

Назва розділу	Назва і зміст підрозділу
1. Вступ	1.1. Назва проекту: «Модель прогнозування соціально-економічних показників розвитку держави» 1.2. Галузь застосування: Аналітичні системи державних органів і фінансових установ для короткострокового прогнозування макроекономічних показників (ВВП, безробіття, інфляція, чисельність населення)
2. Підстава для розробки	2.1. Освітній курс за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології. 2.2. Завдання на дипломну роботу бакалавра, затверджено наказом ХНУ імені В. Н. Каразіна №4101-5/962 від 16.04.2025 р. (представить як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3. Призначення розробки	3.1. Мета: Підвищення точності та швидкості отримання економічних прогнозів шляхом автоматизованого застосування моделей XGBoost і Bidirectional LSTM. 3.2. Призначення: Програмний модуль, що приймає історичні дані *World Development Indicators* і формує прогноз на 1–4 квартали наперед з оцінкою похибки. 3.3. Початкові дані для розробки: CSV-вибірки WDI (1960–2023 рр.); офіційна документація XGBoost, TensorFlow/Keras; технічні вимоги замовника щодо формату вихідних звітів.
4. Технічні вимоги до програмного виробу	4.1. Функціональні характеристики: • імпорт і очищення WDI-даних; • автоматичне формування лагових ознак; • навчання/оновлення моделей XGBoost і LSTM; • обчислення RMSE, MAE, побудова графіків; • експорт прогнозу у CSV/JSON. 4.2. Надійність: Резервне зберігання моделей; відновлення стану після збоїв; unit-тести покриттям ≥ 75 % критичного коду. 4.3. Умови експлуатації: Linux-сервер або Windows-10 Pro; Python 3.10+; мінімум 8 GB RAM, CPU ≥ 4 ядер; GPU не обов'язковий, але бажаний для прискорення LSTM. 4.4. Технічні засоби: PyCharm IDE; бібліотеки: pandas, scikit-learn, xgboost, tensorflow, matplotlib. 4.5. Сумісність: Експорт даних у форматах CSV, JSON, Excel; REST-API сумісне з будь-якою BI-системою (Power BI, Tableau).

	4.6. Маркування та упаковка: Архів “forecast_system.zip” з підкаталогами */src*, */models*, */docs* та файлом README.md. 4.7. Транспортування та зберігання: Передача через захищений репозиторій Git; резервні копії на сервері замовника (мінімум два незалежні носії). 4.8. Спеціальні вимоги: Шифрування конфіденційних датасетів (AES-256); журналювання дій користувача; відповідність GDP
5. Вимоги до програмної документації.	Технічний опис (PDF); інструкція користувача; автоматизовані скрипти встановлення (setup.sh / requirements.txt).
6. Техніко-економічні показники	Середній час побудови прогнозу < 10 с.; зменшення RMSE інфляції ≥ 80 % порівняно з базовою ARIMA; окупність витрат на розробку ≤ 12 міс.
7. Стадії і етапи розробки	1. Огляд літератури з проблематики прогнозування соціально-економічних показників. 2. Порівняльний аналіз традиційних статистичних методів і методів машинного навчання для прогнозування соціально-економічних показників (LSTM, XGBoost тощо). 3. Збирання та підготовка даних з World Development Indicators (WDI), включно з очищенням і масштабуванням. 4. Розробка та проєктування LSTM-моделі, яка здатна враховувати часові залежності та виявляти довготривалі тренди у показниках. 5. Побудова моделі XGBoost з урахуванням особливостей часових рядів, включно зі створенням lag-ознак і налаштуванням гіперпараметрів. 6. Тестування моделей на обраних метриках (RMSE, MAE, MAPE), первинний аналіз результатів 7. Поглиблений аналіз результатів, визначення ефективності та особливостей кожної моделі. 8. Підготовка рекомендацій з удосконалення моделей (логарифмування, розширення фіч, тюнінг гіперпараметрів). 9. Підготовка та оформлення звітних матеріалів.
8. Порядок контролю і приймання	Функціональне тестування за чек-листом замовника; перевірка метрик на контрольному наборі; підписання акту приймання-передачі після успішного проходження всіх тестів.

Виконавець
студент групи КУ-41
Пономаренко В.В.
_____

Замовник
к.т.н., доцент ЗВО
Стрілець В.Є.
_____

Додаток В

Програма і методика випробувань програмного вироби

«Модель прогнозування соціально-економічних показників розвитку держави»

1 Об'єкт випробувань

1.1 Назва: Модель прогнозування соціально-економічних показників розвитку держави.

1.2 Область застосування: Аналітичні системи державних органів і фінансових установ для короткострокового прогнозування макроекономічних показників (ВВП, безробіття, інфляція, чисельність населення).

2. Мета випробувань

Загальна мета: підвищення точності та швидкості отримання економічних прогнозів шляхом автоматизованого застосування моделей XGBoost і Bidirectional LSTM.

Специфічні цілі: оптимізувати підготовку даних WDI, підібрати та налаштувати гіперпараметри XGBoost, розробити Bidirectional LSTM, забезпечити відтворюваність експериментів, порівняти точність моделей.

3. Загальні положення

3.1 Підстави для проведення випробувань

Вимоги акредитаційної комісії університету.

3.2 Місце і тривалість випробувань

Лабораторії факультету, один місяць.

3.3 Обсяг випробувань

Повний цикл випробувань всіх модулів системи.

3.4 Організації, які беруть участь у випробуваннях

ХНУ імені В. Н. Каразіна, факультет комп'ютерних наук.

4. Вимоги до програми або програмного виробу

4.1. Функціональні характеристики:

- імпорт і очищення WDI-даних;
- автоматичне формування лагових ознак;
- навчання/оновлення моделей XGBoost і LSTM;
- обчислення RMSE, MAE, побудова графіків;
- експорт прогнозу у CSV/JSON.

4.2. Надійність: Резервне зберігання моделей; відновлення стану після збоїв; unit-тести покриттям ≥ 75 % критичного коду.

4.3. Умови експлуатації: Linux-сервер або Windows-10 Pro; Python 3.10+; мінімум 8 GB RAM, CPU ≥ 4 ядер; GPU не обов'язковий, але бажаний для прискорення LSTM.

4.4. Технічні засоби: PyCharm IDE; бібліотеки: pandas, scikit-learn, xgboost, tensorflow, matplotlib.

4.5. Сумісність: Експорт даних у форматах CSV, JSON, Excel; REST-API сумісне з будь-якою BI-системою (Power BI, Tableau).

4.6. Маркування та упаковка: Архів "forecast_system.zip" з підкаталогами */src*, */models*, */docs* та файлом README.md.

4.7. Транспортування та зберігання: Передача через захищений репозиторій Git; резервні копії на сервері замовника (мінімум два незалежні носії).

4.8. Спеціальні вимоги: Шифрування конфіденційних датасетів (AES-256); журналювання дій користувача; відповідність GDP

5. Вимоги до програмної документації

Технічний опис (PDF); інструкція користувача; автоматизовані скрипти встановлення (setup.sh / requirements.txt).

6. Засоби і порядок випробувань

6.1 Засоби випробувань

IDE для запуску, налагодження й профілювання коду, Виконання скриптів; ізоляція залежностей, Інтерактивний аналіз метрик і візуалізація графіків.

6.2 Порядок проведення випробувань

Тест №1 – GDP — XGBoost Plot

Мета тесту: Перевірити коректність побудови та збереження графіка *gdp_xgb.png* (фактичне vs прогноз XGBoost).

1. Запустити python forecast_run.py --save-plots.
2. Переконатися, що у каталозі */plots* з'явився файл *gdp_xgb.png*.
3. Засобом matplotlib.testing.compare зчитати фігуру, перевірити:
   • на полотні присутні **дві** криві;
   • назви ліній у легенді — *Actual* і *XGB*;
   • довжина масиву даних дорівнює кількості точок тестового періоду (n_test).

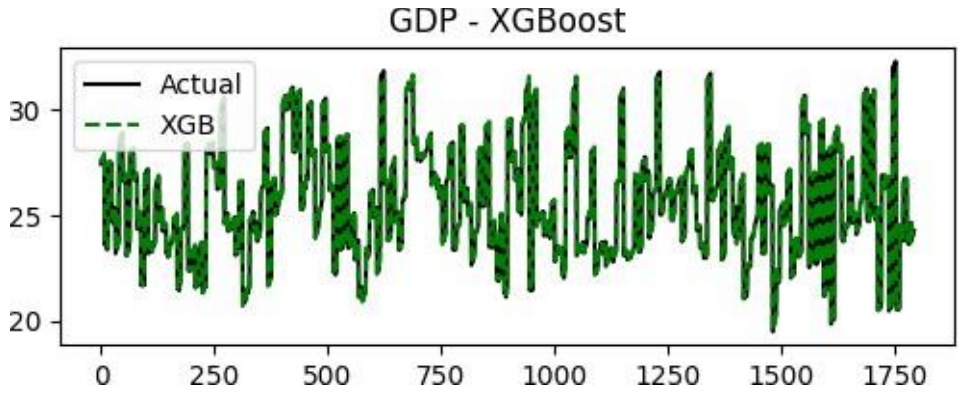

Рисунок В.1 – Порівняння фактичних та прогнозних значень ВВП за моделлю XGBoost

Тест №2 – GDP — LSTM Plot

Мета тесту: Перевірити побудову графіка *gdp_lstm.png* (фактичне vs прогноз LSTM) і допустимий рівень помилки.

1. Запустити скрипт із параметром model=lstm (або обрати графік *gdp_lstm.png* після повного запуску).

2. Переконатися, що файл існує та містить **дві** криві (*Actual*, *LSTM*).

3. Додатково обчислити RMSE для пари «факт–LSTM» і переконатися, що RMSE ≤ 1.0.

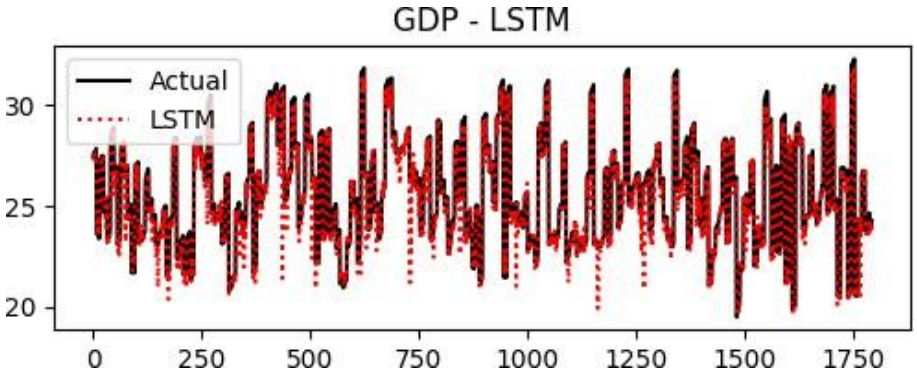

Рисунок В.2 – Порівняння фактичних та прогнозних значень ВВП за моделлю LSTM (тестовий період)

Тест №3 – Таблиця метрик RMSE/MAE

Мета тесту: Переконатися, що зведена таблиця показників точності сформована й містить усі 4 показники для обох моделей

1. Після завершення скрипта перевірити наявність файлу metrics_summary.csv (або друку таблички в консоль).

2. Засобом pandas.read_csv() зчитати таблицю.

3. Перевірити:

- є 4 рядки (*GDP, Unemployment, Inflation, Population*);
- присутні стовпці *RMSE_LSTM, RMSE_XGB, MAE_LSTM, MAE_XGB*;
- жодне значення не є NaN.

Таблиця В.1

Порівняння метрик якості моделей

Показник	RMSE (LSTM)	RMSE (XGB)	MAE (LSTM)	MAE (XGB)
ВВП	0.679	0.128	0.389	0.088
Безробіття	1.982	1.043	1.381	0.664
Інфляція	84.741	21.836	71.066	6.221
Населення	0.511	0.023	0.288	0.016

Висновок: програмний виріб вважається таким, що пройшов випробування в цілому, якщо успішно виконані всі Тести 1, 2, 3. Загалом, тестування показало, що модель працює ефективно та надає достатньо точні результати прогнозування соціально-економічних показників розвитку держав. Це дозволяє зробити висновок про її готовність до використання в реальних умовах або для подальшого доопрацювання.

Виконавець
студент групи КУ-41
Пономаренко В.В.

_____

Додаток Г

Фрагмент лістингу коду

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

!pip install xgboost
from xgboost import XGBRegressor
from sklearn.multioutput import MultiOutputRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping

%matplotlib inline

df_wide = pd.read_csv('WDICSV.csv')
df_long = df_wide.melt(
    id_vars=['Country Name', 'Country Code', 'Indicator Name', 'Indicator
Code'],
    var_name='Year',
    value_name='Value'
)

# Фільтруємо 4 індикатори
gdp_code     = 'NY.GDP.MKTP.CD'     # GDP
unemp_code   = 'SL.UEM.TOTL.ZS'     # Unemployment (%)
infl_code    = 'FP.CPI.TOTL.ZG'     # Inflation (%)
pop_code     = 'SP.POP.TOTL'        # Population

indicators_to_keep = [gdp_code, unemp_code, infl_code, pop_code]
df_long = df_long[df_long['Indicator
Code'].isin(indicators_to_keep)].copy()

# Видаляємо пропуски в основній колонці Value
df_long.dropna(subset=['Value'], inplace=True)

# Перетворюємо рік на числовий формат
df_long['Year'] = pd.to_numeric(df_long['Year'], errors='coerce')
df_long.dropna(subset=['Year'], inplace=True)
df_long['Year'] = df_long['Year'].astype(int)

df_pivot = df_long.pivot_table(
    index=['Country Code','Year'],
    columns='Indicator Code',
```

```
    values='Value'
).reset_index()

df_pivot.columns.name = None
df_pivot.rename(columns={
    gdp_code:   'GDP',
    unemp_code: 'Unemployment',
    infl_code:  'Inflation',
    pop_code:   'Population'
}, inplace=True)

target_cols = ['GDP','Unemployment','Inflation','Population']
print("Перші 5 рядків після pivot:")
print(df_pivot.head())

def create_lags_by_country(df, group_col='Country Code', target_cols=None,
lag=1):
    df_lagged_list = []
    for country, grp in df.groupby(group_col):
        grp_sorted = grp.sort_values('Year').copy()
        for col in target_cols:
            grp_sorted[f'{col}_lag{lag}'] = grp_sorted[col].shift(lag)
        df_lagged_list.append(grp_sorted)
    df_lagged = pd.concat(df_lagged_list, axis=0)
    df_lagged.dropna(inplace=True)
    df_lagged.reset_index(drop=True, inplace=True)
    return df_lagged

cols_for_lag = target_cols
df_lagged = create_lags_by_country(df_pivot, group_col='Country Code',
                                   target_cols=cols_for_lag, lag=1)

# Вхідні ознаки XGBoost
feature_cols_xgb = [f'{col}_lag1' for col in target_cols]

df_lagged.dropna(subset=feature_cols_xgb + target_cols, inplace=True)

train_df_xgb = df_lagged[df_lagged['Year'] <= 2010].copy()
test_df_xgb  = df_lagged[df_lagged['Year'] > 2010].copy()

X_train_xgb = train_df_xgb[feature_cols_xgb].values
y_train_xgb = train_df_xgb[target_cols].values
X_test_xgb  = test_df_xgb[feature_cols_xgb].values
y_test_xgb  = test_df_xgb[target_cols].values

scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()

X_train_xgb_scaled = scaler_X.fit_transform(X_train_xgb)
y_train_xgb_scaled = scaler_y.fit_transform(y_train_xgb)
```

```
X_test_xgb_scaled  = scaler_X.transform(X_test_xgb)
y_test_xgb_scaled  = scaler_y.transform(y_test_xgb)

xgb_reg = XGBRegressor(
    n_estimators=300,
    max_depth=4,
    learning_rate=0.05,
    subsample=0.8,
    colsample_bytree=0.8,
    random_state=42
)

xgb_model = MultiOutputRegressor(xgb_reg)
xgb_model.fit(X_train_xgb_scaled, y_train_xgb_scaled)

# Прогноз
y_pred_xgb_scaled = xgb_model.predict(X_test_xgb_scaled)
y_pred_xgb = scaler_y.inverse_transform(y_pred_xgb_scaled)

print("\n=== РЕЗУЛЬТАТИ XGBOOST ===")
for i, col in enumerate(target_cols):
    rmse = np.sqrt(mean_squared_error(y_test_xgb[:, i], y_pred_xgb[:, i]))
    mae  = mean_absolute_error(y_test_xgb[:, i], y_pred_xgb[:, i])
    print(f"{col}: RMSE={rmse:.2f}, MAE={mae:.2f}")

def create_sequences_for_country(df, window_size, target_cols):
    X_seq, y_seq, years_seq = [], [], []
    df_sorted = df.sort_values('Year').reset_index(drop=True)
    for i in range(len(df_sorted) - window_size):
        X_seq.append(df_sorted[target_cols].iloc[i:i+window_size].values)
        y_seq.append(df_sorted[target_cols].iloc[i+window_size].values)
        years_seq.append(df_sorted['Year'].iloc[i+window_size])
    return X_seq, y_seq, years_seq

# Копія основного pivot
df_pivot_lstm = df_pivot.copy()

# Видаляємо NaN у потрібних колонках, інакше послідовності будуть з NaN
df_pivot_lstm.dropna(subset=target_cols, inplace=True)

window_size = 3
X_all, y_all, years_all = [], [], []

for country, grp in df_pivot_lstm.groupby('Country Code'):
    grp = grp.sort_values('Year').reset_index(drop=True)
    if len(grp) <= window_size:
        continue
    X_seq, y_seq, yrs = create_sequences_for_country(grp, window_size,
target_cols)
```

```
    X_all.extend(X_seq)
    y_all.extend(y_seq)
    years_all.extend(yrs)

X_all = np.array(X_all)
y_all = np.array(y_all)
years_all = np.array(years_all)

train_idx = (years_all <= 2010)
test_idx  = (years_all > 2010)

X_train_lstm = X_all[train_idx]
y_train_lstm = y_all[train_idx]
X_test_lstm  = X_all[test_idx]
y_test_lstm  = y_all[test_idx]

print("\nСтруктура LSTM:")
print("X_train_lstm:", X_train_lstm.shape, "y_train_lstm:",
y_train_lstm.shape)
print("X_test_lstm:", X_test_lstm.shape, "y_test_lstm:",
y_test_lstm.shape)

scaler_X_lstm = MinMaxScaler()
scaler_y_lstm = MinMaxScaler()

nsamples, t, nfeat = X_train_lstm.shape

X_train_flat = X_train_lstm.reshape(-1, nfeat)
X_train_flat = np.nan_to_num(X_train_flat, nan=0.0, posinf=0.0,
neginf=0.0)
X_train_flat_scaled = scaler_X_lstm.fit_transform(X_train_flat)
X_train_lstm_scaled = X_train_flat_scaled.reshape(nsamples, t, nfeat)

nsamples_test = X_test_lstm.shape[0]
X_test_flat = X_test_lstm.reshape(-1, nfeat)
X_test_flat = np.nan_to_num(X_test_flat, nan=0.0, posinf=0.0, neginf=0.0)
X_test_flat_scaled = scaler_X_lstm.transform(X_test_flat)
X_test_lstm_scaled = X_test_flat_scaled.reshape(nsamples_test, t, nfeat)

y_train_lstm_scaled = scaler_y_lstm.fit_transform(y_train_lstm)
y_test_lstm_scaled  = scaler_y_lstm.transform(y_test_lstm)

model_lstm = Sequential()
model_lstm.add(LSTM(64, return_sequences=True, activation='tanh',
                    input_shape=(X_train_lstm_scaled.shape[1],
X_train_lstm_scaled.shape[2])))
model_lstm.add(Dropout(0.3))
model_lstm.add(LSTM(64, activation='tanh'))
model_lstm.add(Dropout(0.3))
model_lstm.add(Dense(len(target_cols)))  # 4 виходи
```

```
# Наприклад, менший lr (бо 1e-5 дуже повільний)
optimizer_lstm = Adam(learning_rate=5e-4, clipnorm=1.0)
model_lstm.compile(loss='mse', optimizer=optimizer_lstm)

early_stop = EarlyStopping(monitor='val_loss', patience=10,
restore_best_weights=True)

history_lstm = model_lstm.fit(
    X_train_lstm_scaled,
    y_train_lstm_scaled,
    epochs=100,
    batch_size=16,
    validation_split=0.2,
    callbacks=[early_stop],
    verbose=1
)

# Прогноз на тесті
y_pred_lstm_scaled = model_lstm.predict(X_test_lstm_scaled)
y_pred_lstm = scaler_y_lstm.inverse_transform(y_pred_lstm_scaled)

print("\n=== РЕЗУЛЬТАТИ LSTM ===")
for i, col in enumerate(target_cols):
    rmse = np.sqrt(mean_squared_error(y_test_lstm[:, i], y_pred_lstm[:,
i]))
    mae  = mean_absolute_error(y_test_lstm[:, i], y_pred_lstm[:, i])
    print(f"{col}: RMSE={rmse:.2f}, MAE={mae:.2f}")

print("\nПОРІВНЯННЯ XGBoost vs LSTM (Test set):")
for i, col in enumerate(target_cols):
    rmse_xgb = np.sqrt(mean_squared_error(y_test_xgb[:, i], y_pred_xgb[:,
i]))
    mae_xgb  = mean_absolute_error(y_test_xgb[:, i], y_pred_xgb[:, i])

    rmse_lstm = np.sqrt(mean_squared_error(y_test_lstm[:, i],
y_pred_lstm[:, i]))
    mae_lstm  = mean_absolute_error(y_test_lstm[:, i], y_pred_lstm[:, i])

    print(f"\n[{col}]")
    print(f"  XGBoost: RMSE={rmse_xgb:.2f}, MAE={mae_xgb:.2f}")
    print(f"  LSTM:    RMSE={rmse_lstm:.2f}, MAE={mae_lstm:.2f}")

plt.figure(figsize=(14,10))

for i, col in enumerate(target_cols):
    plt.subplot(2,2,i+1)

    # LSTM
    plt.plot(
```

```
        range(len(y_test_lstm)),
        y_test_lstm[:, i], 'k-o', label='Actual (LSTM test)' if i==0 else
'Actual',
        markersize=3
    )
    plt.plot(
        range(len(y_pred_lstm)),
        y_pred_lstm[:, i], 'g--', label='LSTM pred',
        linewidth=1
    )

    # XGBoost
    plt.plot(
        range(len(y_test_xgb)),
        y_test_xgb[:, i], 'b-o', label='Actual (XGB test)',
        markersize=3, alpha=0.5
    )
    plt.plot(
        range(len(y_pred_xgb)),
        y_pred_xgb[:, i], 'r--', label='XGB pred',
        linewidth=1, alpha=0.7
    )

    plt.title(col)
    plt.xlabel('Test index')
    plt.ylabel(col)
    plt.legend()
    plt.grid(True)

plt.tight_layout()
plt.show()
```